LRA_project_code1

June 26, 2022

[1]:
```python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import missingno as mss
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
```

[2]:
```python
import os
PROJECT_ROOT_DIR = "."
IMAGES_PATH = os.path.join(PROJECT_ROOT_DIR, "images")
os.makedirs(IMAGES_PATH, exist_ok=True)

def save_fig(fig_id, tight_layout=True, fig_extension="png", resolution=300):
    path = os.path.join(IMAGES_PATH, fig_id + "." + fig_extension)
    print("Saving figure", fig_id)
    if tight_layout:
        plt.tight_layout()
    plt.savefig(path, format=fig_extension, dpi=resolution)
```

## Data cleaning

[3]:
```python
df=pd.read_csv("data\\Beijing.csv")
df=df.iloc[:21,1:26]
df.index=pd.date_range(start="20001231",end="20201231", freq="Y")
economy=pd.DataFrame()
society=pd.DataFrame()
ecology=pd.DataFrame()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 21 entries, 2000-12-31 to 2020-12-31
Freq: A-DEC
Data columns (total 25 columns):
 #   Column                                                             Non-Null
```

```
Count  Dtype
---  ------
--------------  -----
 0   Employment personnel in urban units                               14 non-
null    float64
 1   Employments personnel in urban: first industry                    13 non-
null    float64
 2   Per capita disposable income of urban residents                   14 non-
null    float64
 3   urban population                                                  16 non-
null    float64
 4   GDP                                                               20 non-
null    float64
 5   Primary Industry                                                  20 non-
null    float64
 6   Secondary Industry                                                20 non-
null    float64
 7   tertiary industry                                                 20 non-
null    object
 8   Primary Industry proportion                                       20 non-
null    float64
 9   Secondary Industry proportion                                     20 non-
null    float64
 10  tertiary industry proportion                                      20 non-
null    float64
 11  Local general public budget revenue                               16 non-
null    float64
 12  rainfall                                                          21 non-
null    float64
 13  urban per capita consumption: education,culture and entertainment  18 non-
null    float64
 14  Grain product output                                              21 non-
null    float64
 15  population                                                        19 non-
null    float64
 16  rural net income                                                  11 non-
null    float64
 17  rural average living area                                         13 non-
null    float64
 18  consumption expense per capita                                    18 non-
null    float64
 19  forest coverage                                                   16 non-
null    float64
 20  unemployment rate                                                 19 non-
null    float64
 21  health workers                                                    21 non-
null    float64
 22  rural pupolation                                                  16 non-
```

```
null     float64
 23  water and soil erosion                                             16 non-
null     float64
 24  rural workers                                                      8 non-
null      float64
dtypes: float64(24), object(1)
memory usage: 4.3+ KB
```

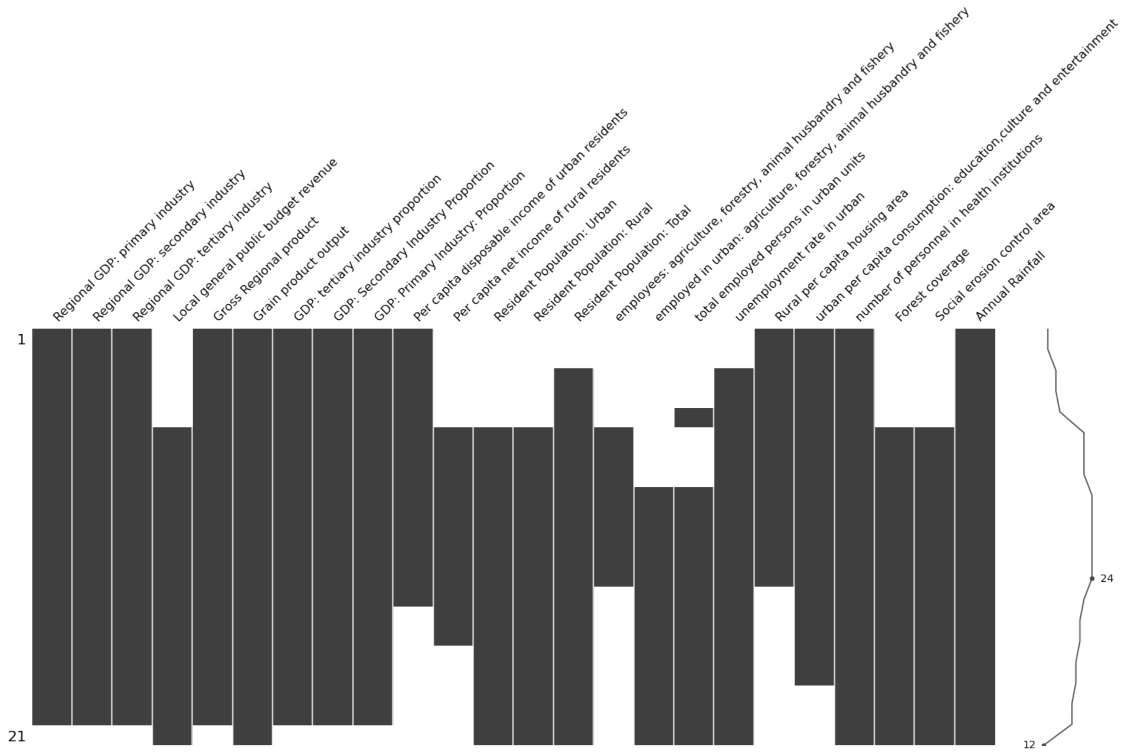
[4]:

```
economy[['Regional GDP: primary industry', 'Regional GDP: secondary industry',␣
↪'Regional GDP: tertiary industry', 'Local general public budget revenue',
       'Gross Regional product', 'Grain product output', 'GDP: tertiary␣
↪industry proportion',
       'GDP: Secondary Industry Proportion', 'GDP: Primary Industry:␣
↪Proportion', 'Per capita disposable income of urban residents', 'Per capita␣
↪net income of rural residents']]=df[["Primary Industry","Secondary␣
↪Industry","tertiary industry",
                                                                              ␣
↪                                                                            ␣
↪                          "Local general public budget revenue","GDP","Grain␣
↪product output",
                                                                              ␣
↪                                                                            ␣
↪                          "tertiary industry proportion","Secondary Industry␣
↪proportion",
                                                                              ␣
↪                                                                            ␣
↪                          "Primary Industry proportion","Per capita disposable␣
↪income of urban residents",
                                                                              ␣
↪                                                                            ␣
↪                          "rural net income"
                                                                              ␣
↪                                                                            ␣
↪                          ]]

ecology[["Forest coverage","Social erosion control area","Annual␣
↪Rainfall"]]=df[["forest coverage","water and soil erosion","rainfall"]]
# df=df.iloc[:,1:]

society=df[["urban population","rural pupolation","population",
                                                            "rural␣
↪workers","Employments personnel in urban: first industry",
                                                            "Employment␣
↪personnel in urban units","unemployment rate","rural average living area",
                                                            "urban per capita␣
↪consumption: education,culture and entertainment","health workers"]]
```

```
# df.index=range(21)
# df2
# df=df.join(df2,how="inner")
society.columns=['Resident Population: Urban', 'Resident Population: Rural',␣
 ↪'Resident Population: Total',
       'employees: agriculture, forestry, animal husbandry and fishery',␣
 ↪'employed in urban: agriculture, forestry, animal husbandry and fishery',␣
 ↪'total employed persons in urban units',
       'unemployment rate in urban',
       'Rural per capita housing area', 'urban per capita consumption:␣
 ↪education,culture and entertainment',
         'number of personnel in health institutions']
```

[5]:
```
df2=economy.join(society,how="outer").join(ecology,how="outer")
# df2
mss.matrix(df2)
save_fig("missing")
```

```
Saving figure missing
```



[6]:
```
df2.to_csv("data\\beijing_cleaned")
# rainFall=pd.Series([5067.22,5911.62,5446.02,6911.42,5693.22,5120.18,5938.
 ↪28,6872.71,6065.10],index=range(9,18))
```

```
# rainFall.name="rainFall"
```

[7]:
```
# df=df.join(rainFall,how="outer")
# df=df.astype(float)
```

[8]:
```
0
# df.columns=['Regional GDP: primary industry', 'Regional GDP: secondary␣
↪industry', 'Regional GDP: tertiary industry', 'Local general public budget␣
↪revenue',
#         'Gross Regional product', 'Grain product output', 'Beijing: GDP', 'GDP:
↪ tertiary industry proportion',
#         'GDP: Secondary Industry Proportion', 'GDP: Primary Industry:␣
↪Proportion', 'Per capita disposable income of urban residents', 'Per capita␣
↪net income of rural residents',
#         'Resident Population: Proportion: Urban', 'Resident Population:␣
↪Proportion: Rural', 'Resident Population: Total', 'Urban Population␣
↪Proportion',
#         'employees: agriculture, forestry, animal husbandry and fishery',␣
↪'employed in urban: agriculture, forestry, animal husbandry and fishery',␣
↪'total employed persons in urban units',
#         'the number of registered unemployed in urban areas', 'the number of␣
↪rural employees', "Engel's coefficient of urban residents",
#         'Rural per capita housing area', 'urban per capita consumption:␣
↪education,culture and entertainment', 'number of personnel in health␣
↪institutions',
#         'Forest coverage', 'Soil erosion control area', 'annual rainFall']
#
# mss.matrix(df2)
# save_fig("missing")
```

[8]:
```
0
```

[9]:
```
df2=pd.read_csv("data/shanxi_cleaned.csv")
df2=df2.iloc[:,1:25]
df2.index=pd.date_range(start="20001231",end="20201231", freq="Y")
df2
```

[9]:
```
            Regional GDP: primary industry  Regional GDP: secondary industry  \
2000-12-31                             NaN                               NaN
2001-12-31                             NaN                               NaN
2002-12-31                             NaN                               NaN
2003-12-31                             NaN                               NaN
2004-12-31                          276.30                           1919.40
2005-12-31                          262.42                           2353.16
2006-12-31                          276.77                           2748.33
2007-12-31                          269.68                           3438.58
```

```
2008-12-31                                  302.48                               4265.77
2009-12-31                                  477.59                               3993.80
2010-12-31                                  554.48                               5234.00
2011-12-31                                  641.42                               6635.26
2012-12-31                                  698.32                               6731.56
2013-12-31                                  773.81                               6792.68
2014-12-31                                  788.89                               6293.91
2015-12-31                                  783.16                               5194.27
2016-12-31                                  784.78                               5028.99
2017-12-31                                  719.16                               6778.89
2018-12-31                                  740.75                               7074.46
2019-12-31                                  824.72                               7453.09
2020-12-31                                     NaN                                   NaN

            Regional GDP: tertiary industry  \
2000-12-31                              NaN
2001-12-31                              NaN
2002-12-31                              NaN
2003-12-31                              NaN
2004-12-31                          1375.67
2005-12-31                          1563.94
2006-12-31                          1727.44
2007-12-31                          2025.09
2008-12-31                          2370.48
2009-12-31                          2886.92
2010-12-31                          3412.38
2011-12-31                          3960.87
2012-12-31                          4682.95
2013-12-31                          5035.75
2014-12-31                          5678.69
2015-12-31                          6789.06
2016-12-31                          7236.64
2017-12-31                          8030.37
2018-12-31                          8142.92
2019-12-31                          8748.87
2020-12-31                              NaN

            Local general public budget revenue  Gross Regional product  \
2000-12-31                                  NaN                     NaN
2001-12-31                                  NaN                     NaN
2002-12-31                                  NaN                     NaN
2003-12-31                                  NaN                     NaN
2004-12-31                                  NaN                 3571.37
2005-12-31                            3683437.0                 4230.53
2006-12-31                            5833752.0                 4878.61
2007-12-31                            5978870.0                 6024.45
2008-12-31                            7480047.0                 7315.40
```

```
2009-12-31                         8058279.0               7358.31
2010-12-31                         9696652.0               9200.86
2011-12-31                        12134300.0              11237.55
2012-12-31                        15163780.0              12112.83
2013-12-31                        17016227.0              12665.25
2014-12-31                        18206400.0              12761.49
2015-12-31                        16423500.0              12766.49
2016-12-31                        15570000.0              13050.41
2017-12-31                        18670022.0              15528.42
2018-12-31                        22927000.0              15958.13
2019-12-31                        23475600.0              17026.68
2020-12-31                        22965700.0                   NaN

            Grain product output  GDP: tertiary industry proportion  \
2000-12-31              853.3500                                NaN
2001-12-31              692.1000                                NaN
2002-12-31              925.5400                                NaN
2003-12-31              958.8700                                NaN
2004-12-31             1062.0000                                NaN
2005-12-31              978.0000                            37.4000
2006-12-31             1073.3300                            36.4000
2007-12-31             1007.0500                            35.3000
2008-12-31             1028.0000                            34.2000
2009-12-31              942.0000                            39.2000
2010-12-31             1085.1000                            37.1000
2011-12-31             1193.0000                            35.2000
2012-12-31             1274.1000                            38.6600
2013-12-31             1312.8000                            40.0000
2014-12-31             1330.7800                            44.4986
2015-12-31             1259.5700                            53.1788
2016-12-31             1318.5000                            55.4514
2017-12-31             1355.0954                            51.7140
2018-12-31             1380.4000                            53.4400
2019-12-31             1361.8000                                NaN
2020-12-31             1424.2700                                NaN

            GDP: Secondary Industry Proportion  \
2000-12-31                                 NaN
2001-12-31                                 NaN
2002-12-31                                 NaN
2003-12-31                                 NaN
2004-12-31                                 NaN
2005-12-31                             56.3000
2006-12-31                             57.8000
2007-12-31                             60.0000
2008-12-31                             61.5000
2009-12-31                             54.3000
```

```
2010-12-31                                56.9000
2011-12-31                                59.0000
2012-12-31                                55.5700
2013-12-31                                53.9000
2014-12-31                                49.3196
2015-12-31                                40.6868
2016-12-31                                38.5351
2017-12-31                                43.6547
2018-12-31                                45.1500
2019-12-31                                    NaN
2020-12-31                                    NaN

            GDP: Primary Industry: Proportion  \
2000-12-31                                NaN
2001-12-31                                NaN
2002-12-31                                NaN
2003-12-31                                NaN
2004-12-31                                NaN
2005-12-31                             6.3000
2006-12-31                             5.8000
2007-12-31                             4.7000
2008-12-31                             4.4000
2009-12-31                             6.5000
2010-12-31                             6.0000
2011-12-31                             5.7000
2012-12-31                             5.7700
2013-12-31                             6.1000
2014-12-31                             6.1818
2015-12-31                             6.1345
2016-12-31                             6.0134
2017-12-31                             4.6312
2018-12-31                             4.4000
2019-12-31                                NaN
2020-12-31                                NaN

            Per capita disposable income of urban residents  …  \
2000-12-31                                        4724.1100  …
2001-12-31                                        5391.0500  …
2002-12-31                                        6234.3600  …
2003-12-31                                        7005.0300  …
2004-12-31                                        7902.8600  …
2005-12-31                                        8913.9100  …
2006-12-31                                       10027.7000  …
2007-12-31                                       11564.9500  …
2008-12-31                                       13119.0500  …
2009-12-31                                       13996.5500  …
2010-12-31                                       15647.6600  …
```

```
2011-12-31                                          18123.8700  …
2012-12-31                                          20411.7100  …
2013-12-31                                          22455.6342  …
2014-12-31                                           8809.4365  …
2015-12-31                                                 NaN  …
2016-12-31                                                 NaN  …
2017-12-31                                                 NaN  …
2018-12-31                                                 NaN  …
2019-12-31                                                 NaN  …
2020-12-31                                                 NaN  …

            employees: agriculture, forestry, animal husbandry and fishery  \
2000-12-31                                                NaN
2001-12-31                                                NaN
2002-12-31                                                NaN
2003-12-31                                                NaN
2004-12-31                                                NaN
2005-12-31                                             637.44
2006-12-31                                             635.68
2007-12-31                                             633.92
2008-12-31                                             637.85
2009-12-31                                             631.62
2010-12-31                                             632.44
2011-12-31                                             643.43
2012-12-31                                             640.68
2013-12-31                                                NaN
2014-12-31                                                NaN
2015-12-31                                                NaN
2016-12-31                                                NaN
2017-12-31                                                NaN
2018-12-31                                                NaN
2019-12-31                                                NaN
2020-12-31                                                NaN

            employed in urban: agriculture, forestry, animal husbandry and
fishery  \
2000-12-31                                                NaN
2001-12-31                                                NaN
2002-12-31                                                NaN
2003-12-31                                                NaN
2004-12-31                                                NaN
2005-12-31                                                NaN
2006-12-31                                                NaN
2007-12-31                                                NaN
2008-12-31                                                3.5
2009-12-31                                                2.6
2010-12-31                                                3.2
```

```
2011-12-31                                                3.2
2012-12-31                                                2.8
2013-12-31                                                2.2
2014-12-31                                                2.0
2015-12-31                                                1.8
2016-12-31                                                1.7
2017-12-31                                                1.6
2018-12-31                                                1.4
2019-12-31                                                1.0
2020-12-31                                                1.3

            total employed persons in urban units  unemployment rate in urban  \
2000-12-31                                    NaN                         NaN
2001-12-31                                    NaN                         NaN
2002-12-31                                    NaN                         3.4
2003-12-31                                    NaN                         3.0
2004-12-31                                  362.0                         3.1
2005-12-31                                    NaN                         3.0
2006-12-31                                    NaN                         3.2
2007-12-31                                    NaN                         3.2
2008-12-31                                  375.2                         3.3
2009-12-31                                  385.8                         3.9
2010-12-31                                  394.4                         3.6
2011-12-31                                  409.7                         3.5
2012-12-31                                  436.0                         3.3
2013-12-31                                  464.0                         3.1
2014-12-31                                  452.1                         3.4
2015-12-31                                  440.3                         3.5
2016-12-31                                  430.6                         3.5
2017-12-31                                  428.7                         3.4
2018-12-31                                  425.8                         3.3
2019-12-31                                  441.1                         2.7
2020-12-31                                  442.6                         3.1

            Rural per capita housing area  \
2000-12-31                        21.5742
2001-12-31                        22.2748
2002-12-31                        22.7000
2003-12-31                        22.9400
2004-12-31                        23.2580
2005-12-31                        24.1454
2006-12-31                        24.9614
2007-12-31                        25.7968
2008-12-31                        26.5200
2009-12-31                        27.9700
2010-12-31                        28.2452
2011-12-31                        29.9220
```

```
2012-12-31                          30.6081
2013-12-31                              NaN
2014-12-31                              NaN
2015-12-31                              NaN
2016-12-31                              NaN
2017-12-31                              NaN
2018-12-31                              NaN
2019-12-31                              NaN
2020-12-31                              NaN

            urban per capita consumption: education,culture and entertainment  \
2000-12-31                                           501.7800
2001-12-31                                           567.8500
2002-12-31                                           781.8000
2003-12-31                                           799.3500
2004-12-31                                           901.4000
2005-12-31                                           932.5300
2006-12-31                                          1007.9200
2007-12-31                                          1054.0500
2008-12-31                                          1041.9100
2009-12-31                                          1070.6000
2010-12-31                                          1229.6800
2011-12-31                                          1419.4300
2012-12-31                                          1506.2000
2013-12-31                                          2065.4400
2014-12-31                                          2026.5227
2015-12-31                                          2207.9274
2016-12-31                                          2438.9628
2017-12-31                                          2559.4273
2018-12-31                                                NaN
2019-12-31                                                NaN
2020-12-31                                                NaN

            number of personnel in health institutions  Forest coverage  \
2000-12-31                                       19.45              NaN
2001-12-31                                       20.05              NaN
2002-12-31                                       16.67              NaN
2003-12-31                                       16.99              NaN
2004-12-31                                       17.47             14.1
2005-12-31                                       17.39             14.1
2006-12-31                                       17.90             14.1
2007-12-31                                       17.51             14.1
2008-12-31                                       19.12             14.1
2009-12-31                                       26.44             18.0
2010-12-31                                       27.60             18.0
2011-12-31                                       27.16             18.0
2012-12-31                                       27.95             18.0
```

```
2013-12-31                                        28.39            18.0
2014-12-31                                        28.93            20.5
2015-12-31                                        29.49            20.5
2016-12-31                                        31.13            20.5
2017-12-31                                        31.90            20.5
2018-12-31                                        33.09            20.5
2019-12-31                                        34.17            20.5
2020-12-31                                        35.14            20.5

            Social erosion control area  Annual Rainfall
2000-12-31                          NaN            419.3
2001-12-31                          NaN            298.0
2002-12-31                          NaN            419.4
2003-12-31                          NaN            525.4
2004-12-31                          NaN            377.2
2005-12-31                    5184.5100            274.7
2006-12-31                    5424.3100            424.8
2007-12-31                    5639.4400            535.4
2008-12-31                    4969.3200            355.3
2009-12-31                    5093.8770            625.1
2010-12-31                    5352.4950            376.6
2011-12-31                    5560.6182            496.6
2012-12-31                    5290.6206            427.8
2013-12-31                    5475.6944            487.3
2014-12-31                    5667.9700            428.7
2015-12-31                    5846.3889            403.6
2016-12-31                    6171.7600            528.4
2017-12-31                    6484.7900            521.2
2018-12-31                    6798.5100            364.6
2019-12-31                    7086.5400            312.6
2020-12-31                    7395.7400            547.0

[21 rows x 24 columns]
```

0.0.1 missing data completion

```
[10]: dfFilled=df2.fillna(method="ffill").fillna(method="bfill")

      dfFilled
```

```
[10]:             Regional GDP: primary industry  Regional GDP: secondary industry  \
      2000-12-31                          276.30                           1919.40
      2001-12-31                          276.30                           1919.40
      2002-12-31                          276.30                           1919.40
      2003-12-31                          276.30                           1919.40
      2004-12-31                          276.30                           1919.40
      2005-12-31                          262.42                           2353.16
```

```
2006-12-31                          276.77                         2748.33
2007-12-31                          269.68                         3438.58
2008-12-31                          302.48                         4265.77
2009-12-31                          477.59                         3993.80
2010-12-31                          554.48                         5234.00
2011-12-31                          641.42                         6635.26
2012-12-31                          698.32                         6731.56
2013-12-31                          773.81                         6792.68
2014-12-31                          788.89                         6293.91
2015-12-31                          783.16                         5194.27
2016-12-31                          784.78                         5028.99
2017-12-31                          719.16                         6778.89
2018-12-31                          740.75                         7074.46
2019-12-31                          824.72                         7453.09
2020-12-31                          824.72                         7453.09

            Regional GDP: tertiary industry  \
2000-12-31                          1375.67
2001-12-31                          1375.67
2002-12-31                          1375.67
2003-12-31                          1375.67
2004-12-31                          1375.67
2005-12-31                          1563.94
2006-12-31                          1727.44
2007-12-31                          2025.09
2008-12-31                          2370.48
2009-12-31                          2886.92
2010-12-31                          3412.38
2011-12-31                          3960.87
2012-12-31                          4682.95
2013-12-31                          5035.75
2014-12-31                          5678.69
2015-12-31                          6789.06
2016-12-31                          7236.64
2017-12-31                          8030.37
2018-12-31                          8142.92
2019-12-31                          8748.87
2020-12-31                          8748.87

            Local general public budget revenue  Gross Regional product  \
2000-12-31                            3683437.0                 3571.37
2001-12-31                            3683437.0                 3571.37
2002-12-31                            3683437.0                 3571.37
2003-12-31                            3683437.0                 3571.37
2004-12-31                            3683437.0                 3571.37
2005-12-31                            3683437.0                 4230.53
2006-12-31                            5833752.0                 4878.61
```

```
2007-12-31                          5978870.0                6024.45
2008-12-31                          7480047.0                7315.40
2009-12-31                          8058279.0                7358.31
2010-12-31                          9696652.0                9200.86
2011-12-31                         12134300.0               11237.55
2012-12-31                         15163780.0               12112.83
2013-12-31                         17016227.0               12665.25
2014-12-31                         18206400.0               12761.49
2015-12-31                         16423500.0               12766.49
2016-12-31                         15570000.0               13050.41
2017-12-31                         18670022.0               15528.42
2018-12-31                         22927000.0               15958.13
2019-12-31                         23475600.0               17026.68
2020-12-31                         22965700.0               17026.68

            Grain product output  GDP: tertiary industry proportion  \
2000-12-31              853.3500                            37.4000
2001-12-31              692.1000                            37.4000
2002-12-31              925.5400                            37.4000
2003-12-31              958.8700                            37.4000
2004-12-31             1062.0000                            37.4000
2005-12-31              978.0000                            37.4000
2006-12-31             1073.3300                            36.4000
2007-12-31             1007.0500                            35.3000
2008-12-31             1028.0000                            34.2000
2009-12-31              942.0000                            39.2000
2010-12-31             1085.1000                            37.1000
2011-12-31             1193.0000                            35.2000
2012-12-31             1274.1000                            38.6600
2013-12-31             1312.8000                            40.0000
2014-12-31             1330.7800                            44.4986
2015-12-31             1259.5700                            53.1788
2016-12-31             1318.5000                            55.4514
2017-12-31             1355.0954                            51.7140
2018-12-31             1380.4000                            53.4400
2019-12-31             1361.8000                            53.4400
2020-12-31             1424.2700                            53.4400

            GDP: Secondary Industry Proportion  \
2000-12-31                             56.3000
2001-12-31                             56.3000
2002-12-31                             56.3000
2003-12-31                             56.3000
2004-12-31                             56.3000
2005-12-31                             56.3000
2006-12-31                             57.8000
2007-12-31                             60.0000
```

```
2008-12-31                                61.5000
2009-12-31                                54.3000
2010-12-31                                56.9000
2011-12-31                                59.0000
2012-12-31                                55.5700
2013-12-31                                53.9000
2014-12-31                                49.3196
2015-12-31                                40.6868
2016-12-31                                38.5351
2017-12-31                                43.6547
2018-12-31                                45.1500
2019-12-31                                45.1500
2020-12-31                                45.1500

            GDP: Primary Industry: Proportion  \
2000-12-31                                6.3000
2001-12-31                                6.3000
2002-12-31                                6.3000
2003-12-31                                6.3000
2004-12-31                                6.3000
2005-12-31                                6.3000
2006-12-31                                5.8000
2007-12-31                                4.7000
2008-12-31                                4.4000
2009-12-31                                6.5000
2010-12-31                                6.0000
2011-12-31                                5.7000
2012-12-31                                5.7700
2013-12-31                                6.1000
2014-12-31                                6.1818
2015-12-31                                6.1345
2016-12-31                                6.0134
2017-12-31                                4.6312
2018-12-31                                4.4000
2019-12-31                                4.4000
2020-12-31                                4.4000

            Per capita disposable income of urban residents  …  \
2000-12-31                                        4724.1100  …
2001-12-31                                        5391.0500  …
2002-12-31                                        6234.3600  …
2003-12-31                                        7005.0300  …
2004-12-31                                        7902.8600  …
2005-12-31                                        8913.9100  …
2006-12-31                                       10027.7000  …
2007-12-31                                       11564.9500  …
2008-12-31                                       13119.0500  …
```

```
2009-12-31                                        13996.5500  …
2010-12-31                                        15647.6600  …
2011-12-31                                        18123.8700  …
2012-12-31                                        20411.7100  …
2013-12-31                                        22455.6342  …
2014-12-31                                         8809.4365  …
2015-12-31                                         8809.4365  …
2016-12-31                                         8809.4365  …
2017-12-31                                         8809.4365  …
2018-12-31                                         8809.4365  …
2019-12-31                                         8809.4365  …
2020-12-31                                         8809.4365  …

            employees: agriculture, forestry, animal husbandry and fishery  \
2000-12-31                                             637.44
2001-12-31                                             637.44
2002-12-31                                             637.44
2003-12-31                                             637.44
2004-12-31                                             637.44
2005-12-31                                             637.44
2006-12-31                                             635.68
2007-12-31                                             633.92
2008-12-31                                             637.85
2009-12-31                                             631.62
2010-12-31                                             632.44
2011-12-31                                             643.43
2012-12-31                                             640.68
2013-12-31                                             640.68
2014-12-31                                             640.68
2015-12-31                                             640.68
2016-12-31                                             640.68
2017-12-31                                             640.68
2018-12-31                                             640.68
2019-12-31                                             640.68
2020-12-31                                             640.68

            employed in urban: agriculture, forestry, animal husbandry and
fishery  \
2000-12-31                                                3.5
2001-12-31                                                3.5
2002-12-31                                                3.5
2003-12-31                                                3.5
2004-12-31                                                3.5
2005-12-31                                                3.5
2006-12-31                                                3.5
2007-12-31                                                3.5
2008-12-31                                                3.5
```

```
2009-12-31                                            2.6
2010-12-31                                            3.2
2011-12-31                                            3.2
2012-12-31                                            2.8
2013-12-31                                            2.2
2014-12-31                                            2.0
2015-12-31                                            1.8
2016-12-31                                            1.7
2017-12-31                                            1.6
2018-12-31                                            1.4
2019-12-31                                            1.0
2020-12-31                                            1.3

            total employed persons in urban units  unemployment rate in urban  \
2000-12-31                                  362.0                         3.4
2001-12-31                                  362.0                         3.4
2002-12-31                                  362.0                         3.4
2003-12-31                                  362.0                         3.0
2004-12-31                                  362.0                         3.1
2005-12-31                                  362.0                         3.0
2006-12-31                                  362.0                         3.2
2007-12-31                                  362.0                         3.2
2008-12-31                                  375.2                         3.3
2009-12-31                                  385.8                         3.9
2010-12-31                                  394.4                         3.6
2011-12-31                                  409.7                         3.5
2012-12-31                                  436.0                         3.3
2013-12-31                                  464.0                         3.1
2014-12-31                                  452.1                         3.4
2015-12-31                                  440.3                         3.5
2016-12-31                                  430.6                         3.5
2017-12-31                                  428.7                         3.4
2018-12-31                                  425.8                         3.3
2019-12-31                                  441.1                         2.7
2020-12-31                                  442.6                         3.1

            Rural per capita housing area  \
2000-12-31                        21.5742
2001-12-31                        22.2748
2002-12-31                        22.7000
2003-12-31                        22.9400
2004-12-31                        23.2580
2005-12-31                        24.1454
2006-12-31                        24.9614
2007-12-31                        25.7968
2008-12-31                        26.5200
2009-12-31                        27.9700
```

```
2010-12-31                        28.2452
2011-12-31                        29.9220
2012-12-31                        30.6081
2013-12-31                        30.6081
2014-12-31                        30.6081
2015-12-31                        30.6081
2016-12-31                        30.6081
2017-12-31                        30.6081
2018-12-31                        30.6081
2019-12-31                        30.6081
2020-12-31                        30.6081

            urban per capita consumption: education,culture and entertainment  \
2000-12-31                                                         501.7800
2001-12-31                                                         567.8500
2002-12-31                                                         781.8000
2003-12-31                                                         799.3500
2004-12-31                                                         901.4000
2005-12-31                                                         932.5300
2006-12-31                                                        1007.9200
2007-12-31                                                        1054.0500
2008-12-31                                                        1041.9100
2009-12-31                                                        1070.6000
2010-12-31                                                        1229.6800
2011-12-31                                                        1419.4300
2012-12-31                                                        1506.2000
2013-12-31                                                        2065.4400
2014-12-31                                                        2026.5227
2015-12-31                                                        2207.9274
2016-12-31                                                        2438.9628
2017-12-31                                                        2559.4273
2018-12-31                                                        2559.4273
2019-12-31                                                        2559.4273
2020-12-31                                                        2559.4273

            number of personnel in health institutions  Forest coverage  \
2000-12-31                                       19.45             14.1
2001-12-31                                       20.05             14.1
2002-12-31                                       16.67             14.1
2003-12-31                                       16.99             14.1
2004-12-31                                       17.47             14.1
2005-12-31                                       17.39             14.1
2006-12-31                                       17.90             14.1
2007-12-31                                       17.51             14.1
2008-12-31                                       19.12             14.1
2009-12-31                                       26.44             18.0
2010-12-31                                       27.60             18.0
```

```
2011-12-31                                       27.16            18.0
2012-12-31                                       27.95            18.0
2013-12-31                                       28.39            18.0
2014-12-31                                       28.93            20.5
2015-12-31                                       29.49            20.5
2016-12-31                                       31.13            20.5
2017-12-31                                       31.90            20.5
2018-12-31                                       33.09            20.5
2019-12-31                                       34.17            20.5
2020-12-31                                       35.14            20.5

            Social erosion control area  Annual Rainfall
2000-12-31                    5184.5100            419.3
2001-12-31                    5184.5100            298.0
2002-12-31                    5184.5100            419.4
2003-12-31                    5184.5100            525.4
2004-12-31                    5184.5100            377.2
2005-12-31                    5184.5100            274.7
2006-12-31                    5424.3100            424.8
2007-12-31                    5639.4400            535.4
2008-12-31                    4969.3200            355.3
2009-12-31                    5093.8770            625.1
2010-12-31                    5352.4950            376.6
2011-12-31                    5560.6182            496.6
2012-12-31                    5290.6206            427.8
2013-12-31                    5475.6944            487.3
2014-12-31                    5667.9700            428.7
2015-12-31                    5846.3889            403.6
2016-12-31                    6171.7600            528.4
2017-12-31                    6484.7900            521.2
2018-12-31                    6798.5100            364.6
2019-12-31                    7086.5400            312.6
2020-12-31                    7395.7400            547.0

[21 rows x 24 columns]
```

[11]:
```
dfFilled.index=pd.date_range(start="20001231",end="20201231", freq="Y")
dfFilled.iloc[:,0]=dfFilled.iloc[:,0]/dfFilled['Resident Population: Total']
scaleNeeded=[0,1,2,3,4,5,11,12,14,15,16,20]
for i in scaleNeeded:
    dfFilled.iloc[:,i]=dfFilled.iloc[:,i]/dfFilled['Resident Population: Total']
# dfFilled['Resident Population: Total']=dfFilled['Resident Population: Total']/
 ↪dfFilled['Resident Population: Total'][0]
dfFilled['Resident Population: Total']=dfFilled['Resident Population: Total']/
 ↪dfFilled['Resident Population: Total'][0]
```

[12]:
```
dfFilled.to_csv("data\\shanxiFilled.csv")
```

0.1 entropy

[13]:
```
dfFilled.columns
posNeg=pd.Series([1,1, 1, 1,1, 1,1, 1,1,  1,1,   1, -1,1, 1,1,  1,-1, 1, ␣
↪1,1,1,    1,1,   ],index=dfFilled.columns)
```

[13]:
```

```

[14]:
```
dfFilled=pd.read_csv("data/beijingFilled.csv").iloc[:,1:]
```

[15]:
```
dfBackup=dfFilled.copy()
economy=dfBackup.iloc[:,range(11)]
social=dfBackup.iloc[:,range(11,21)]
ecology=dfBackup.iloc[:,range(21,24)]
```

[16]:
```
def proportion(df):
    df_pro=df
    for i in range(len(df.columns)):
        Sum=np.sum(df.iloc[:,i])
        df_pro.iloc[:,i]=df.iloc[:,i].apply(lambda x: x/Sum if x!=0 else 1e-6)
    return df_pro


def entropy(df,PosNeg):
    dfScaled=df.copy()
    tmp=df.iloc[:,2]
    for i in range(len(df.columns)):
        max=np.max(df.iloc[:,i])
        min=np.min(df.iloc[:,i])
        if PosNeg[i]==1:
            dfScaled.iloc[:,i]=(df.iloc[:,i]-min)/(max-min)
        else:
            dfScaled.iloc[:,i]=(max-df.iloc[:,i])/(max-min)
    df_pro=proportion(dfScaled)
    k=1/np.log(len(df.columns))
    e=[]
    for i in range(len(df.columns)):
        ei=(-k)*sum(df_pro.iloc[:,i]*np.log(df_pro.iloc[:,i]))
        e.append(ei)
    d=1-np.array(e)
    sumD=np.sum(d)
    weight=[]
    for i in range(len(d)):
        weight.append(d[i]/sumD)
    weight=pd.Series(weight,index=df.columns)
    return weight
```

```
[17]: dfFilled.columns
```

```
[17]: Index(['Regional GDP: primary industry', 'Regional GDP: secondary industry',
              'Regional GDP: tertiary industry',
              'Local general public budget revenue', 'Gross Regional product',
              'Grain product output', 'GDP: tertiary industry proportion',
              'GDP: Secondary Industry Proportion',
              'GDP: Primary Industry: Proportion',
              'Per capita disposable income of urban residents',
              'Per capita net income of rural residents',
              'Resident Population: Urban', 'Resident Population: Rural',
              'Resident Population: Total',
              'employees: agriculture, forestry, animal husbandry and fishery',
              'employed in urban: agriculture, forestry, animal husbandry and fishery',
              'total employed persons in urban units', 'unemployment rate in urban',
              'Rural per capita housing area',
              'urban per capita consumption: education,culture and entertainment',
              'number of personnel in health institutions', 'Forest coverage',
              'Social erosion control area', 'Annual Rainfall'],
             dtype='object')
```

```
[18]: weightEconomy=entropy(economy,np.ones(11).astype(int))
      weightSociety=entropy(social, [1, -1,1, 1,1,  1,-1, 1,  1,1])
      weightEcology=entropy(ecology,[1,1,1])
```

```
[19]: weightEcology
```

```
[19]: Forest coverage                0.273304
      Social erosion control area    0.327285
      Annual Rainfall                0.399411
      dtype: float64
```

```
[20]: weightSociety
```

```
[20]: Resident Population: Urban
      0.108311
      Resident Population: Rural
      0.093069
      Resident Population: Total
      0.081166
      employees: agriculture, forestry, animal husbandry and fishery
      0.034905
      employed in urban: agriculture, forestry, animal husbandry and fishery
      0.130004
      total employed persons in urban units
      0.124064
      unemployment rate in urban
```

```
0.130902
Rural per capita housing area
0.126666
urban per capita consumption: education,culture and entertainment
0.105583
number of personnel in health institutions
0.065331
dtype: float64
```

[21]:
```
weightEconomy
```

[21]:
```
Regional GDP: primary industry                     0.110749
Regional GDP: secondary industry                   0.116638
Regional GDP: tertiary industry                    0.085794
Local general public budget revenue                0.047943
Gross Regional product                             0.093953
Grain product output                               0.101270
GDP: tertiary industry proportion                  0.112208
GDP: Secondary Industry Proportion                 0.104690
GDP: Primary Industry: Proportion                  0.077444
Per capita disposable income of urban residents    0.106251
Per capita net income of rural residents           0.043061
dtype: float64
```

[22]:
```
pd.DataFrame([weightEconomy,weightSociety,weightEcology]).
↪to_csv("weight_subsystem.csv")
```

[23]:
```
# economyHarmony=[]
# for i in range(24):
#     if i%9==0:
#         pass
#         # input()
#     else:
#         tmp=int(input())
#         economyHarmony.append(tmp)
#
economyHarmony=[0.4021718, 0.3784322, 0.3673585, 0.3564703, 0.3824495, 0.
↪3642708, 0.3647985,
                0.3898616, 0.4322923, 0.4481538, 0.4780799, 0.5283633, 0.
↪5624757, 0.5932242,
                0.5925471, 0.5848710, 0.4528035, 0.6091901, 0.6278512, 0.
↪6274901, 0.6225538]
societyHarmony=[0.4197542, 0.4211727, 0.4324124, 0.4231431, 0.4580266, 0.
↪2304821, 0.2806699,
                0.3724445, 0.3716443, 0.5000562, 0.5647700, 0.5839717, 0.
↪6113522, 0.6871906,
                0.7204494, 0.7396891, 0.7530385, 0.7274197, 0.7570196, 0.
↪7414146, 0.6998066]
ecologyHarmony=[0.2115706, 0.1870682, 0.2732383, 0.2562908, 0.2878388, 0.
↪1066354, 0.1259026,
                0.1567999, 0.1850360, 0.2602829, 0.4287615, 0.3694112, 0.
↪6426935, 0.4977946,
                0.4213142, 0.5907259, 0.7756946, 0.6545896, 0.7983126, 0.
↪8688439, 0.8893232]
# harmonySystems=pd.
↪DataFrame([economyHarmony,societyHarmony,ecologyHarmony],columns=dfFilled.
↪index,index=["economy","society","ecology"]).transpose()
harmonySystems=pd.read_csv("data/beijing_subscore.csv").iloc[:, 1:]
harmonySystems
```

[23]:

```
    beijing.eco_score  beijing.econ_score  beijing.socia_score
0            0.051081            0.373890             0.488130
1            0.020105            0.350036             0.523304
2            0.050119            0.341000             0.529499
3            0.121689            0.326081             0.509486
4            0.159206            0.376465             0.494757
5            0.089175            0.389437             0.396298
6            0.011743            0.384009             0.466884
7            0.189537            0.377969             0.461132
8            0.345376            0.407811             0.500710
9            0.238915            0.444110             0.594754
10           0.390031            0.457016             0.627459
11           0.602323            0.505913             0.633176
12           0.627949            0.542641             0.679640
13           0.496150            0.564534             0.735801
14           0.406509            0.556323             0.722544
15           0.607903            0.546490             0.778643
16           0.829147            0.554544             0.787712
17           0.760272            0.562804             0.817056
18           0.752995            0.596885             0.808766
19           0.666721            0.605680             0.731919
20           0.802603            0.603391             0.608561
```

[24]:

```
harmonyBack=harmonySystems.copy()
print(entropy(harmonyBack,[1,1,1]))
```

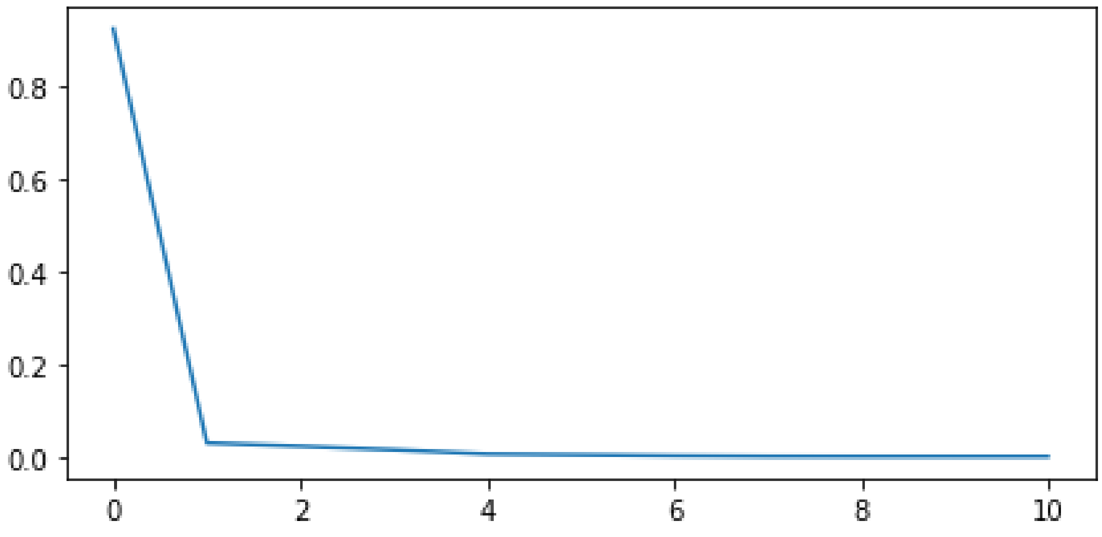
```
beijing.eco_score      0.320144
beijing.econ_score     0.333288
beijing.socia_score    0.346568
dtype: float64
```

0.2 pca

[25]:
```
from sklearn import preprocessing
from sklearn import decomposition
scaler=preprocessing.StandardScaler()
dfScaled=pd.DataFrame(scaler.fit_transform(dfFilled),index=dfFilled.
 ↪index,columns=dfFilled.columns)
economyScaled=dfScaled.iloc[:,range(11)]
socialScaled=dfScaled.iloc[:,range(11,21)]
ecologyScaled=dfScaled.iloc[:,range(21,24)]
```

[26]:
```
fig, ax = plt.subplots(figsize=(6, 3))
economyScaled
PCA=decomposition.PCA()
PCA.fit(economyScaled)
plt.plot(PCA.explained_variance_ratio_)
save_fig("economyVariance")
```

```
Saving figure economyVariance
```



[27]:
```
PCA.explained_variance_ratio_
PCA.explained_variance_
```

[27]:
```
array([1.06715527e+01, 3.34522768e-01, 2.53982009e-01, 1.66092202e-01,
       6.10127975e-02, 4.02500509e-02, 1.75044084e-02, 3.96895495e-03,
       8.02402434e-04, 3.08426519e-04, 3.30539468e-06])
```

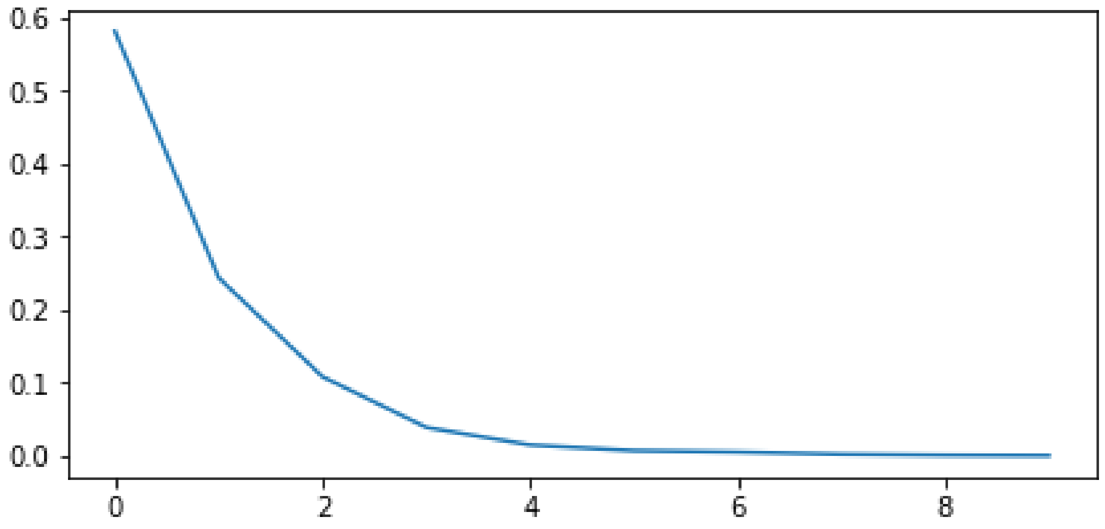
[28]:
```
economyPCA=decomposition.PCA(n_components=3)
economyPCA.fit(economyScaled)
```

```
pd.DataFrame(economyPCA.components_,columns=economyScaled.columns).transpose()
```

```
[28]:                                                        0         1         2
      Regional GDP: primary industry                  0.296631  0.030504 -0.542817
      Regional GDP: secondary industry               -0.309687 -0.108253 -0.181970
      Regional GDP: tertiary industry                -0.307809  0.142866  0.235138
      Local general public budget revenue            -0.308043  0.048581  0.273061
      Gross Regional product                         -0.309340  0.093589  0.165194
      Grain product output                            0.269376 -0.843821  0.359373
      GDP: tertiary industry proportion              -0.307529  0.015360  0.081329
      GDP: Secondary Industry Proportion              0.309379  0.154124  0.021736
      GDP: Primary Industry: Proportion               0.292901  0.243413  0.538920
      Per capita disposable income of urban residents -0.299405 -0.368042 -0.294592
      Per capita net income of rural residents       -0.304119 -0.158409  0.040500
```

```
[29]: fig, ax = plt.subplots(figsize=(6, 3))
      PCA.fit(socialScaled)
      plt.plot(PCA.explained_variance_ratio_)
      save_fig("societyVariance")
```

```
Saving figure societyVariance
```



```
[30]: PCA.explained_variance_
```

```
[30]: array([6.10031415e+00, 2.55588948e+00, 1.13479764e+00, 4.05983307e-01,
             1.53129478e-01, 7.04008411e-02, 5.13483710e-02, 2.00079721e-02,
             6.96665955e-03, 1.16210958e-03])
```

```
[31]: societyPCA=decomposition.PCA(n_components=4)
      societyPCA.fit(socialScaled)
      pd.DataFrame(societyPCA.components_,columns=socialScaled.columns).transpose()
```

```
[31]:                                                        0         1  \
      Resident Population: Urban                      0.109426 -0.559483
      Resident Population: Rural                      0.410533 -0.050319
      Resident Population: Total                     -0.405977 -0.036031
      employees: agriculture, forestry, animal husban…  0.400681  0.073404
      employed in urban: agriculture, forestry, anima…  0.288728 -0.023520
      total employed persons in urban units          -0.150521 -0.566391
      unemployment rate in urban                     -0.005480  0.399461
      Rural per capita housing area                  -0.297832  0.402991
      urban per capita consumption: education,culture… -0.407402 -0.037297
      number of personnel in health institutions     -0.365232 -0.181788

                                                             2         3
      Resident Population: Urban                      0.304490 -0.028790
      Resident Population: Rural                      0.061544  0.040923
      Resident Population: Total                     -0.137292 -0.102729
      employees: agriculture, forestry, animal husban…  0.143557  0.081506
      employed in urban: agriculture, forestry, anima… -0.517903 -0.750946
      total employed persons in urban units          -0.039049 -0.250847
      unemployment rate in urban                      0.669540 -0.565218
      Rural per capita housing area                  -0.223539  0.036473
      urban per capita consumption: education,culture… -0.050507 -0.130354
      number of personnel in health institutions      0.305757 -0.125145
```

0.3 FA

```
[32]: from factor_analyzer import FactorAnalyzer
```

```
[33]: economyFactor=FactorAnalyzer(n_factors=3,rotation="promax")
      economyFactor.fit(economyScaled)
      pd.DataFrame(economyFactor.loadings_,index=economyScaled.columns)
```

```
[33]:                                                  0         1         2
      Regional GDP: primary industry           -0.777619 -0.145539  0.057145
      Regional GDP: secondary industry          0.410282  0.220086 -0.431502
      Regional GDP: tertiary industry           0.881826  0.026517 -0.115254
      Local general public budget revenue       0.716620  0.384401  0.069942
      Gross Regional product                    0.804640  0.046561 -0.183336
      Grain product output                     -0.668468 -0.031823  0.176780
      GDP: tertiary industry proportion         0.593422  0.294823 -0.148017
      GDP: Secondary Industry Proportion       -0.475362 -0.208139  0.371534
      GDP: Primary Industry: Proportion        -0.106605 -0.066991  0.855085
      Per capita disposable income of urban residents -0.010344  0.695273 -0.365481
Per capita net income of rural residents        0.332728  0.690787 -0.017702
```

```
[34]: economyFactor.get_factor_variance()
```

```
[34]: (array([3.85877653, 1.31658776, 1.29750489]),
       array([0.35079787, 0.1196898 , 0.11795499]),
       array([0.35079787, 0.47048766, 0.58844265]))
```

```
[35]: societyFactor=FactorAnalyzer(n_factors=4,rotation="promax")
      societyFactor.fit(socialScaled)
      pd.DataFrame(societyFactor.loadings_,index=socialScaled.columns)
```

```
[35]:                                                        0         1  \
      Resident Population: Urban                      -0.165410  0.903335
      Resident Population: Rural                      -0.939948  0.186631
      Resident Population: Total                       0.999065 -0.090834
      employees: agriculture, forestry, animal husban… -0.979677  0.045593
      employed in urban: agriculture, forestry, anima… -0.125602 -0.002850
      total employed persons in urban units            0.555454  0.741544
      unemployment rate in urban                       0.042730 -0.071983
      Rural per capita housing area                    0.600746 -0.751114
      urban per capita consumption: education,culture…  0.986492 -0.039275
      number of personnel in health institutions       0.833831  0.346899

                                                              2         3
      Resident Population: Urban                      -0.005473 -0.070537
      Resident Population: Rural                      -0.006598  0.055244
      Resident Population: Total                      -0.045519  0.037897
      employees: agriculture, forestry, animal husban…  0.078191 -0.022772
      employed in urban: agriculture, forestry, anima…  0.066854  0.948489
      total employed persons in urban units           -0.131681  0.135303
      unemployment rate in urban                       0.996884  0.058595
      Rural per capita housing area                   -0.019916 -0.014724
      urban per capita consumption: education,culture…  0.027880  0.002372
      number of personnel in health institutions       0.214371 -0.171159
```

```
[36]: societyFactor.get_factor_variance()
```

```
[36]: (array([5.22422899, 2.10230572, 1.07097601, 0.96087165]),
       array([0.5224229 , 0.21023057, 0.1070976 , 0.09608716]),
       array([0.5224229 , 0.73265347, 0.83975107, 0.93583824]))
```

```
[37]: economyReduced=economyFactor.transform(economyScaled)
      loading_eco=pd.DataFrame(economyReduced,index=dfFilled.
       ↪index,columns=["economy_1","economy_2","economy_3"])
      loading_eco
```

```
[37]:     economy_1  economy_2  economy_3
      0   -1.146097  -1.031596   2.403891
      1   -1.227271  -0.972382   2.062349
      2   -0.543316  -1.387025   2.047666
      3   -1.915906  -0.831599   0.462359
      4   -0.400125  -1.851677   0.356686
      5   -0.464185  -1.293552   0.651714
      6   -0.625229  -0.616977   0.691831
      7   -0.857287  -0.614689   0.101621
      8   -0.541941  -0.728234  -0.421834
      9   -0.461030  -0.396227  -0.540573
      10  -0.459904  -0.217210  -0.623699
      11  -0.169718   0.326201  -0.508643
      12   0.224453   0.788790  -0.022600
      13  -0.312989   0.841942  -1.217430
      14  -0.281006   1.952463   0.275126
      15   1.438980   1.049676  -0.703854
      16   1.201114   1.115404  -0.749876
      17   0.839163   1.290368  -0.620517
      18   1.958552   0.851127  -1.258394
      19   1.817598   0.915524  -1.307444
      20   1.926143   0.809674  -1.078380
```

```
[38]: societyReduced=societyFactor.transform(socialScaled)
      loading_soc=pd.DataFrame(societyReduced,index=dfFilled.
       ↪index,columns=["social_1","social_2","social_3","social_4"])
      loading_soc
```

```
[38]:     social_1  social_2  social_3  social_4
      0  -1.285975  2.458398 -0.276805  0.732021
      1  -1.441705  1.317625 -0.193729  0.678196
      2  -1.372020  1.175194 -0.257838  0.737533
      3  -1.249786  0.642684 -0.329218  0.690850
      4  -1.092150  0.220536 -0.718540  0.677497
      5  -1.078908 -0.999630  1.673125  0.298547
      6  -0.671860 -1.506350  1.313222  0.140143
      7  -0.781726 -1.754315  0.689961  0.108083
      8  -0.498711 -1.352984  0.825521 -0.229342
      9  -0.214506 -0.726052 -0.445107  0.575837
      10  0.106895 -1.020558 -0.432125  0.408802
      11  0.359328 -0.654714 -0.369283 -0.692775
      12  0.583085 -0.261317 -0.713040 -0.539874
      13  0.812117  0.084114 -1.103718  0.203902
      14  0.797192  0.079839 -0.752762  0.107914
      15  1.000857  0.207443 -0.513053  0.962560
      16  1.054027  0.323085 -0.463841  0.695600
      17  1.194311  0.439936 -0.399815  0.375978
```

```
18  1.232879  0.434388 -0.301641 -0.458385
19  1.329509  0.730731 -0.496813 -3.071285
20  1.217148  0.161946  3.265499 -2.401802
```

0.4 fitting

[39]:
```
reducedX=loading_eco.join(loading_soc,how="outer").
 ↪join(ecologyScaled,how="outer")
reducedX.to_csv("data/reduced.csv")
```

[40]:
```
x_all=dfScaled
```

[41]:
```
y_all=pd.read_csv("data/Province Cohe/beijing_subscore.csv").iloc[:,5]
y_all
```

[41]:
```
0     0.455529
1     0.390076
2     0.453617
3     0.519713
4     0.554010
5     0.486188
6     0.354924
7     0.564384
8     0.642132
9     0.628954
10    0.693490
11    0.760991
12    0.784358
13    0.767973
14    0.738085
15    0.798723
16    0.846396
17    0.840853
18    0.846000
19    0.816649
20    0.817369
Name: beijing.d_cohe, dtype: float64
```

[42]:
```
from sklearn.model_selection import LeaveOneOut

loo=LeaveOneOut()
from sklearn.svm import SVR
def svr_tune(gamma):
    residual=[]
    svr=SVR(kernel="rbf",gamma=gamma,tol=1e-3)
    for train_index, test_index in loo.split(x_all,y_all):
        train_x=np.array(x_all)[train_index]
        train_y=y_all[train_index]
        test_x=np.array(x_all)[test_index]
        test_y=y_all[test_index]
        svr.fit(train_x,train_y)
        predict=svr.predict(test_x)
        residual.append(test_y-predict)
    print(gamma,end=" ")
    print((np.array(residual)*np.array(residual)).mean())
```

[43]:
```
gamma=1e-3
while gamma < 1:
    svr_tune(gamma)
    gamma*=1.2
```

```
0.001 0.007788212646931924
0.0012 0.007332202729308293
0.0014399999999999999 0.007421375911279256
0.0017279999999999997 0.007409849317527246
0.0020735999999999997 0.007480145105263176
0.0024883199999999996 0.0077093912807602175
0.0029859839999999993 0.007819356360266946
0.003583180799999999 0.007890679507234925
0.0042998169599999985 0.007954651579542279
0.005159780351999998 0.008084397461344593
0.0061917364223999976 0.008223140588408992
0.007430083706879997 0.00834995464875752
0.008916100448255996 0.00855255715379539
0.010699320537907194 0.008787540811688952
0.012839184645488633 0.009013152828421061
0.01540702157458636 0.009273262636874358
0.01848842588950363 0.009560043328643148
0.022186111067404354 0.009814284224464028
0.026623333280885224 0.010058844628513874
0.031947999937062266 0.010393643196768543
0.03833759992447472 0.010630432064183671
0.04600511990936966 0.010855207120680246
0.05520614389124359 0.011068358781345116
0.06624737266949231 0.011171895623080208
0.07949684720339077 0.011332303213873436
0.09539621664406893 0.011604288610612594
0.1144754599728827 0.012021626455314951
0.13737055196745923 0.012540219046906545
0.16484466236095108 0.013195208203002668
0.1978135948331413 0.013982177447860133
0.23737631379976953 0.014969688458604559
0.28485157655972343 0.016033699697221366
```

```
0.3418218918716681 0.0171107781080025
0.41018627024600174 0.018353503618560824
0.49222352429520205 0.01967096920467898
0.5906682291542424 0.020967716073805247
0.708801874985091 0.02222481212111298
0.8505622499821092 0.023391388273346858
```

[44]:
```
from xgboost import XGBRegressor
xgb=XGBRegressor()
```

[45]:
```
res=np.zeros(300).reshape((30,10))
```

xgboost

[46]:
```
def xgb_tune(depth,count):
    residual=[]
    xgb=XGBRegressor(max_depth=depth,n_estimators=count,learning_rate=0.2)
    for train_index, test_index in loo.split(x_all,y_all):
        train_x=np.array(x_all)[train_index]
        train_y=y_all[train_index]
        test_x=np.array(x_all)[test_index]
        test_y=y_all[test_index]
        xgb.fit(train_x,train_y)
        predict=xgb.predict(test_x)
        residual.append(test_y-predict)
    print(depth,end=" ")
    print(count,end=" ")
    print(np.sqrt((np.array(residual)*np.array(residual)).mean()))
    res[count//10,depth]=((np.array(residual)*np.array(residual)).mean())
```

[47]:
```
count=50
depth=1
while depth<2:
    while count<160:
        xgb_tune(depth,count)
        count+=10
    depth+=1
    count=10
```

```
1 50 0.047796579817957374
1 60 0.047036154656547396
1 70 0.0465369923823751
1 80 0.04602796036199998
1 90 0.045875148148280136
1 100 0.04570220326904038
1 110 0.04558004526008994
1 120 0.045440154024811566
1 130 0.04526542611738945
```

```
1 140 0.045169343773661‌25
1 150 0.045053528257133914
```

[48]:
```
xgb_tune(1,160)
```

```
1 160 0.044975654439299025
```

lasso

[49]:
```
from sklearn.linear_model import Lasso
def Lasso_tune(lamb):
    residual=[]
    lasso=Lasso(alpha=lamb)
    for train_index, test_index in loo.split(x_all,y_all):
        train_x=np.array(x_all)[train_index]
        train_y=y_all[train_index]
        test_x=np.array(x_all)[test_index]
        test_y=y_all[test_index]
        lasso.fit(train_x,train_y)
        predict=lasso.predict(test_x)
        residual.append(test_y-predict)
    print(lasso.get_params)
    print(lamb, end=' ')
    print(np.sqrt((np.array(residual)*np.array(residual)).mean()))
```

[50]:
```
lamb=0.000001
while lamb<0.005:
    Lasso_tune(lamb)
    lamb*=1.3
```

```
<bound method BaseEstimator.get_params of Lasso(alpha=1e-06)>
1e-06 0.07576094191278188
<bound method BaseEstimator.get_params of Lasso(alpha=1.3e-06)>
1.3e-06 0.07555193798248411
<bound method BaseEstimator.get_params of Lasso(alpha=1.6900000000000001e-06)>
1.6900000000000001e-06 0.07528165971400189
<bound method BaseEstimator.get_params of Lasso(alpha=2.1970000000000003e-06)>
2.1970000000000003e-06 0.07493138751638409
<bound method BaseEstimator.get_params of Lasso(alpha=2.8561000000000005e-06)>
2.8561000000000005e-06 0.07448076677943526
<bound method BaseEstimator.get_params of Lasso(alpha=3.712930000000001e-06)>
3.712930000000001e-06 0.07393854047939258
<bound method BaseEstimator.get_params of Lasso(alpha=4.826809000000001e-06)>
4.826809000000001e-06 0.07327275166385262
<bound method BaseEstimator.get_params of Lasso(alpha=6.274851700000002e-06)>
6.274851700000002e-06 0.07241792646908901
<bound method BaseEstimator.get_params of Lasso(alpha=8.157307210000003e-06)>
8.157307210000003e-06 0.0713514151132859
<bound method BaseEstimator.get_params of Lasso(alpha=1.0604499373000003e-05)>
```

```
1.0604499373000003e-05 0.06999623507793998
<bound method BaseEstimator.get_params of Lasso(alpha=1.3785849184900005e-05)>
1.3785849184900005e-05 0.06828374993786375
<bound method BaseEstimator.get_params of Lasso(alpha=1.7921603940370008e-05)>
1.7921603940370008e-05 0.06613831313541617
<bound method BaseEstimator.get_params of Lasso(alpha=2.329808512248101e-05)>
2.329808512248101e-05 0.06321375768787585
<bound method BaseEstimator.get_params of Lasso(alpha=3.0287510659225314e-05)>
3.0287510659225314e-05 0.05907675167068207
<bound method BaseEstimator.get_params of Lasso(alpha=3.937376385699291e-05)>
3.937376385699291e-05 0.05363590456298271
<bound method BaseEstimator.get_params of Lasso(alpha=5.118589301409078e-05)>
5.118589301409078e-05 0.04718152428418736
<bound method BaseEstimator.get_params of Lasso(alpha=6.654166091831801e-05)>
6.654166091831801e-05 0.04072138104169119
<bound method BaseEstimator.get_params of Lasso(alpha=8.650415919381342e-05)>
8.650415919381342e-05 0.03678172649671562
<bound method BaseEstimator.get_params of Lasso(alpha=0.00011245540695195745)>
0.00011245540695195745 0.03432758148968581
<bound method BaseEstimator.get_params of Lasso(alpha=0.00014619202903754468)>
0.00014619202903754468 0.029462915253132813
<bound method BaseEstimator.get_params of Lasso(alpha=0.00019004963774880808)>
0.00019004963774880808 0.030339462083872327
<bound method BaseEstimator.get_params of Lasso(alpha=0.0002470645290734505)>
0.0002470645290734505 0.03234452081202142
<bound method BaseEstimator.get_params of Lasso(alpha=0.0003211838877954856)>
0.0003211838877954856 0.0355946490863328
<bound method BaseEstimator.get_params of Lasso(alpha=0.00041753905413413134)>
0.00041753905413413134 0.0401816411306427
<bound method BaseEstimator.get_params of Lasso(alpha=0.0005428007703743708)>
0.0005428007703743708 0.04388134405663868
<bound method BaseEstimator.get_params of Lasso(alpha=0.0007056410014866821)>
0.0007056410014866821 0.04557091333052906
<bound method BaseEstimator.get_params of Lasso(alpha=0.0009173333019326867)>
0.0009173333019326867 0.047052910769434325
<bound method BaseEstimator.get_params of Lasso(alpha=0.0011925332925124927)>
0.0011925332925124927 0.04771667063409557
<bound method BaseEstimator.get_params of Lasso(alpha=0.0015502932802662407)>
0.0015502932802662407 0.04781069108680841
<bound method BaseEstimator.get_params of Lasso(alpha=0.002015381264346113)>
0.002015381264346113 0.048156223941460706
<bound method BaseEstimator.get_params of Lasso(alpha=0.0026199956436499467)>
0.0026199956436499467 0.04844361698496451
<bound method BaseEstimator.get_params of Lasso(alpha=0.003405994336744931)>
0.003405994336744931 0.0488424432978247
<bound method BaseEstimator.get_params of Lasso(alpha=0.00442779263776841)>
0.00442779263776841 0.0496628074323516
```

[51]:
```
Lasso_tune(0.0001)
```

```
<bound method BaseEstimator.get_params of Lasso(alpha=0.0001)>
0.0001 0.03545715512675434
```

[52]:
```
scores=np.zeros(21*6).reshape(6,21)
files=os.listdir("data/Province Cohe")
i=0
for file in files:
    dfTemp=pd.read_csv("data/Province Cohe/"+file)
    scores[i]=dfTemp.iloc[:,5]
    i+=1
scores
```

[52]:
```
array([[0.45552898, 0.3900762 , 0.45361671, 0.5197134 , 0.55400966,
        0.48618776, 0.35492405, 0.56438444, 0.64213201, 0.62895446,
        0.69349037, 0.76099126, 0.78435787, 0.76797298, 0.73808481,
        0.79872265, 0.84639593, 0.84085323, 0.84600044, 0.81664943,
        0.81736904],
       [0.53885994, 0.47564929, 0.51613577, 0.53831453, 0.52041488,
        0.52537318, 0.55919491, 0.59164146, 0.66511738, 0.68856132,
        0.67389507, 0.76510105, 0.79934016, 0.69598822, 0.57616775,
        0.78864133, 0.83698494, 0.80415205, 0.77377418, 0.83210821,
        0.86416037],
       [0.52786337, 0.49587946, 0.54388302, 0.57368292, 0.5497771 ,
        0.54560623, 0.56969736, 0.58851392, 0.6317572 , 0.68188572,
        0.68704857, 0.73382291, 0.73884891, 0.57240026, 0.7459659 ,
        0.77927072, 0.81030127, 0.79999141, 0.8037484 , 0.82277344,
        0.84059056],
       [0.47177751, 0.44632999, 0.38721088, 0.52352451, 0.55743887,
        0.57576179, 0.54006033, 0.60225279, 0.61913443, 0.6645448 ,
        0.70746824, 0.71637789, 0.71994217, 0.65286719, 0.71884538,
        0.77620655, 0.82692865, 0.79467425, 0.8282655 , 0.82153902,
        0.8375499 ],
       [0.46163226, 0.36993843, 0.47469974, 0.53329648, 0.47298433,
        0.36532009, 0.52944429, 0.58032494, 0.48954691, 0.69133407,
        0.65905868, 0.74560498, 0.73905328, 0.79271259, 0.79010532,
        0.78256277, 0.82697922, 0.84089938, 0.82149982, 0.83881648,
        0.8997663 ],
       [0.48747889, 0.53628517, 0.37839967, 0.59215953, 0.51278309,
        0.58684375, 0.48970335, 0.46958722, 0.59380818, 0.61539679,
        0.58496832, 0.69358482, 0.78375573, 0.75247897, 0.75629682,
        0.82571712, 0.85521082, 0.82025134, 0.8030192 , 0.79583089,
        0.82511553]])
```

[53]:
```
from pyecharts.charts import Map
from pyecharts import options as opts
```

```
province=[" "," "," "," "," "," "]
data_province = [(province[i], scores[i,0]) for i in range(6)]
china_province = (
    Map()
    .add('', data_province, 'china')
    .set_global_opts(
        title_opts=opts.TitleOpts(title='Coherence 2000'),
        visualmap_opts=opts.VisualMapOpts(
            min_=0.4,
            max_=0.9,
            is_piecewise=True)
    )
    .render(path='images/cohe2000.html')
)
```

0.4.1 regional distribution

[54]:
```
province=[" "," "," "," "," "," "]
data_province = [(province[i], scores[i,5]) for i in range(6)]
china_province = (
    Map()
    .add('', data_province, 'china')
    .set_global_opts(
        title_opts=opts.TitleOpts(title='Coherence 2005'),
        visualmap_opts=opts.VisualMapOpts(
            min_=0.4,
            max_=0.9,
            is_piecewise=True)
    )
    .render(path='images/cohe2005.html')
)
```

[55]:
```
province=[" "," "," "," "," "," "]
data_province = [(province[i], scores[i,10]) for i in range(6)]
china_province = (
    Map()
    .add('', data_province, 'china')
    .set_global_opts(
        title_opts=opts.TitleOpts(title='Coherence 2010'),
        visualmap_opts=opts.VisualMapOpts(
            min_=0.4,
            max_=0.9,
            is_piecewise=True)
    )
    .render(path='images/cohe2010.html')
)
```

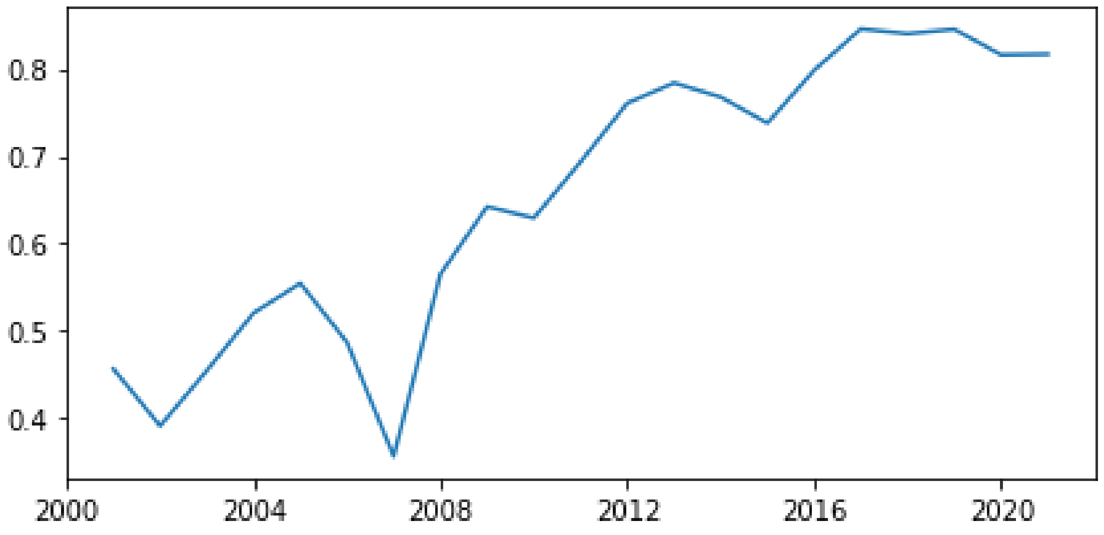
[56]:
```
province=[" "," "," "," "," "," "]
data_province = [(province[i], scores[i,15]) for i in range(6)]
china_province = (
    Map()
    .add('', data_province, 'china')
    .set_global_opts(
        title_opts=opts.TitleOpts(title='Coherence 2015'),
        visualmap_opts=opts.VisualMapOpts(
            min_=0.4,
            max_=0.9,
            is_piecewise=True)
    )
    .render(path='images/cohe2015.html')
)
```

0.4.2 time series forcasting

[57]:
```
beijing_score=pd.Series(np.array(y_all),index = pd.
 ↪date_range(start="20001231",end="20201231", freq="Y"))
from statsmodels.tsa import seasonal
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.stattools import arma_order_select_ic
from statsmodels.tsa.statespace import sarimax
from statsmodels.graphics import tsaplots
from statsmodels.tsa.arima.model import ARIMA
```

[58]:
```
fig, ax = plt.subplots(figsize=(6, 3))
plt.plot(beijing_score)
save_fig("beijing_tsa")
```

```
Saving figure beijing_tsa
```



[59]:
```
adfuller(beijing_score)
adfuller(beijing_score.diff(1).diff(1).dropna())
beijing_stationary=beijing_score.diff(1).diff(1).dropna()
```

[60]:
```
totalArma=arma_order_select_ic(beijing_stationary,max_ar=3,max_ma=3, ic=['bic'])

totalArma.bic
```

```
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
c:\users\sun yc\appdata\local\programs\python\python38\lib\site-
```

```
packages\statsmodels\base\model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

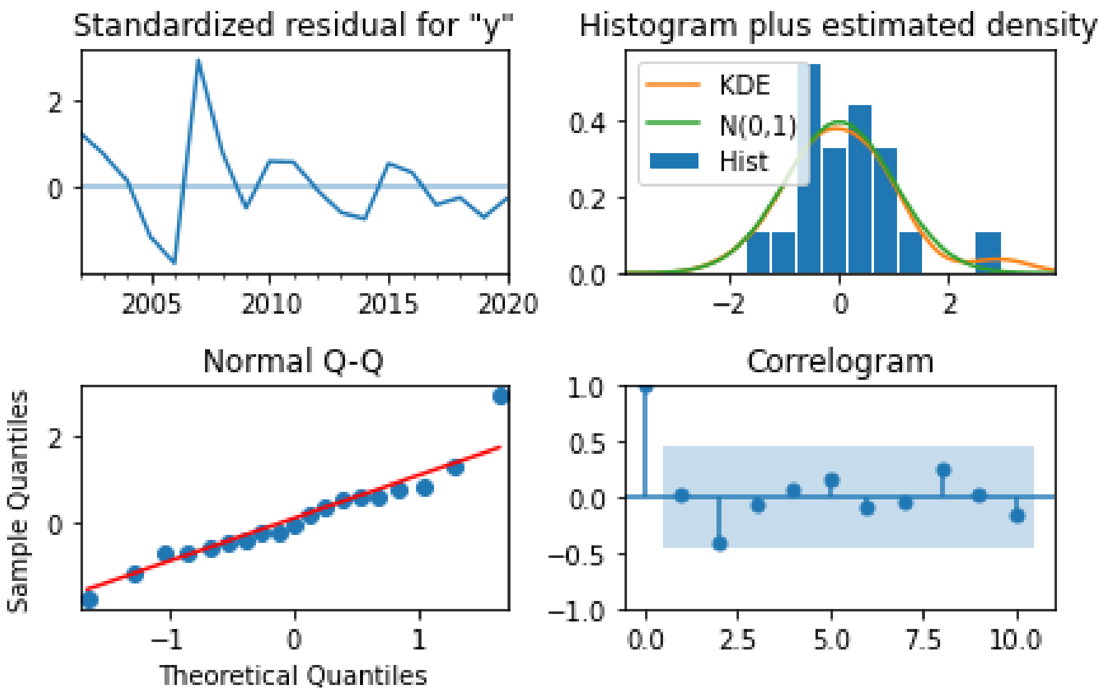
[60]:
```
           0          1          2          3
0 -27.026376 -34.660020 -31.789434 -33.417931
1 -25.550307 -31.720429 -30.686334 -29.870648
2 -28.758318 -32.968151 -29.806523 -26.388928
3 -28.465358 -30.169996 -27.608475 -23.578210
```

[61]:
```
from statsmodels.stats.diagnostic import acorr_ljungbox
acorr_ljungbox(beijing_stationary,lags=range(10))
```

[61]:
```
     lb_stat  lb_pvalue
0  11.144344        NaN
1   1.583627   0.208239
2   5.392095   0.067472
3   5.628994   0.131124
4   5.690580   0.223478
5   6.939511   0.225178
6   8.031078   0.235835
7   8.504410   0.290220
8  11.128491   0.194524
9  11.144344   0.265945
```

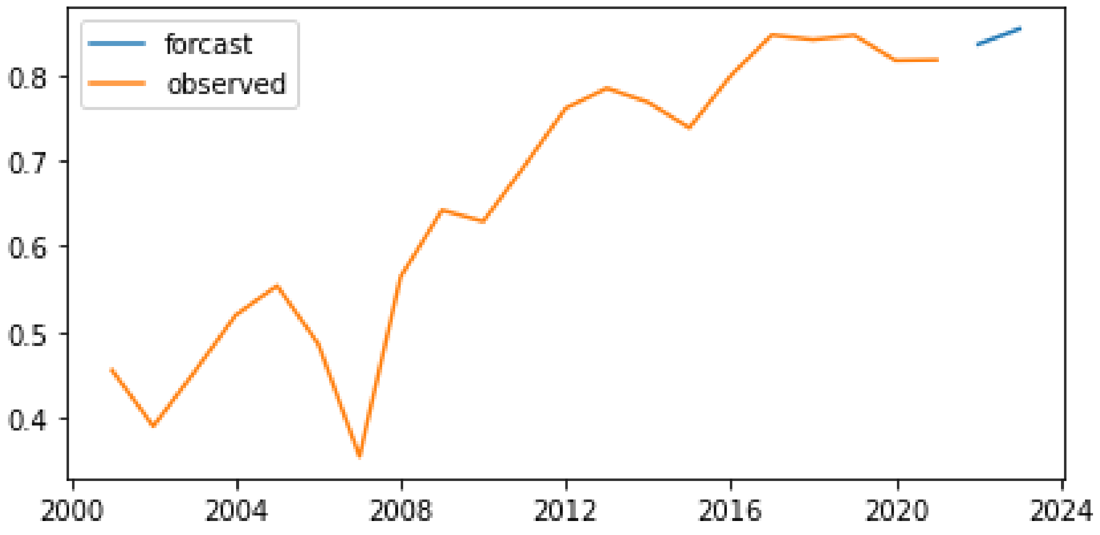
[62]:
```
arima=ARIMA(beijing_score,order=(0,2,1))
tsaFit=arima.fit()
tsaFit.summary()
tsaFit.plot_diagnostics()
save_fig("tsa_diag.png")
```

```
Saving figure tsa_diag.png
```



[63]:
```python
fig, ax = plt.subplots(figsize=(6, 3))
plt.plot(tsaFit.forecast(2),label="forcast")
plt.plot(beijing_score,label="observed")
plt.legend()
save_fig("prediction.png")
```

```
Saving figure prediction.png
```


forcast
observed