In [155…

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
import seaborn as sn
import matplotlib.pyplot as p
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import numpy as np
from numpy import math
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [156…

```python
data_scaled.head()
```

Out[156…

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -1.688194 | -1.283841 | -1.040015 | -1.921745 | -2.012674 | -2.265684 | -2.045899 | 1.924965 | 0.156393 | 0.790569 |
| **1** | -1.599342 | -0.957113 | -1.109625 | -1.632743 | -1.166557 | -2.166566 | -0.640595 | 0.517334 | -0.061440 | 0.790569 |
| **2** | -1.510490 | -1.067904 | -1.072393 | -0.767781 | -1.200402 | -1.848324 | -1.265175 | 0.204528 | 1.463390 | -1.264911 |
| **3** | -1.421637 | -0.957951 | -0.813364 | -0.649612 | -1.196171 | -1.485358 | -1.109030 | 0.830141 | 0.374226 | 0.790569 |
| **4** | -1.332785 | -1.019325 | -0.599586 | -0.073184 | -1.416162 | -1.087486 | -1.733610 | 1.142948 | 1.681223 | 0.790569 |

In [157…

```python
data_scaled['class'] = df.target
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-157-70fe323a6b93> in <module>
----> 1 data_scaled['class'] = df.target

~\anaconda3\lib\site-packages\pandas\core\generic.py in __getattr__(self, name)
   5463             if self._info_axis._can_hold_identifiers_and_holds_name(name):
   5464                 return self[name]
-> 5465             return object.__getattribute__(self, name)
   5466 
   5467     def __setattr__(self, name: str, value) -> None:

AttributeError: 'DataFrame' object has no attribute 'target'
```

In [158…

```python
df = pd.read_csv("d.csv")
print (df)
```

```
    year  agriculture  industries  services   GGRP  \
0   2012        76123      104218    209540  12.00
1   2013       150200       92458    265878  14.00
2   2014       125081       98748    434494  13.92
3   2015       150010      142509    457530  13.93
4   2016       136095      178625    569899  13.41
5   2017       144803      222031    598520  13.49
6   2018       176107      185890    616374  14.76
7   2019       230424      125456    315256  14.25
8   2020       325123      312522    658123  14.58
9   2021       450222      325456    752963  14.25
10  2022       545486      456963    754963  14.47
11  2023       325123      252156    645856  15.47
12  2024       325478      212123    445456  15.47
13  2025       125456      123342    345587  15.68
14  2026       454222      345477    558647  15.47
15  2027       214569      212547    345798  15.98
16  2028       125456      121547    314547  15.34
17  2029       456321      314214    478458  16.58
18  2030       125693      114458    254478  16.47
19  2031       325489      214578    547987  17.45
20  2032       456189      347458    614789  14.68
21  2033       345666      214547    612478  17.15
22  2034       615235      412457    812487  17.59
23  2035       456525      214657    721587  17.56
```

| | | | | | |
|---|---|---|---|---|---|
| 24 | 2036 | 232158 | 121548 | 512478 | 17.98 |
| 25 | 2037 | 154236 | 125478 | 612478 | 17.78 |
| 26 | 2038 | 752963 | 421547 | 824145 | 18.57 |
| 27 | 2039 | 815856 | 521478 | 921457 | 18.54 |
| 28 | 2040 | 641523 | 512478 | 725458 | 18.78 |
| 29 | 2041 | 915236 | 712687 | 947125 | 18.98 |
| 30 | 2042 | 521326 | 412478 | 614798 | 19.12 |
| 31 | 2043 | 456282 | 345789 | 512698 | 19.65 |
| 32 | 2044 | 614236 | 512358 | 725894 | 19.65 |
| 33 | 2045 | 125123 | 111547 | 514654 | 19.47 |
| 34 | 2046 | 215236 | 125145 | 514364 | 19.58 |
| 35 | 2047 | 452456 | 345795 | 812749 | 20.12 |
| 36 | 2048 | 325456 | 245794 | 412789 | 20.36 |
| 37 | 2049 | 912156 | 812547 | 987215 | 20.45 |
| 38 | 2050 | 325456 | 245784 | 812457 | 20.56 |

| | increased GSDP growth(crores) | COA | COI | COS | target | increase growth |
|---|---|---|---|---|---|---|
| 0 | 552854 | 14 | 26 | 60 | 1 | 2.3 |
| 1 | 577902 | 23 | 17 | 59 | 1 | -1.2 |
| 2 | 658325 | 19 | 15 | 66 | 0 | -1.6 |
| 3 | 750050 | 20 | 19 | 61 | 1 | 2.6 |
| 4 | 850596 | 16 | 21 | 67 | 1 | 3.2 |
| 5 | 965355 | 15 | 23 | 62 | 0 | -3.2 |
| 6 | 978373 | 18 | 19 | 63 | 1 | 2.1 |
| 7 | 945231 | 20 | 20 | 60 | 0 | -4.5 |
| 8 | 934634 | 18 | 20 | 62 | 1 | 2.0 |
| 9 | 934653 | 30 | 20 | 50 | 0 | -6.0 |
| 10 | 934876 | 25 | 22 | 53 | 1 | 3.6 |
| 11 | 1012343 | 26 | 14 | 60 | 1 | 2.0 |
| 12 | 1023432 | 29 | 14 | 57 | 0 | -5.6 |
| 13 | 1043432 | 30 | 16 | 54 | 1 | 2.3 |
| 14 | 943234 | 33 | 9 | 58 | 1 | 2.6 |
| 15 | 1043634 | 26 | 15 | 59 | 0 | -3.0 |
| 16 | 954345 | 22 | 14 | 64 | 0 | -5.3 |
| 17 | 1123432 | 31 | 4 | 65 | 0 | -6.0 |
| 18 | 1520236 | 35 | 3 | 62 | 1 | 6.0 |
| 19 | 1134876 | 28 | 11 | 61 | 0 | -4.0 |
| 20 | 912754 | 23 | 14 | 63 | 0 | -3.0 |
| 21 | 1134876 | 28 | 5 | 67 | 0 | -2.3 |
| 22 | 1256345 | 25 | 16 | 59 | 1 | 6.0 |
| 23 | 1298689 | 27 | 18 | 55 | 0 | -4.0 |
| 24 | 1287957 | 26 | 23 | 51 | 1 | 3.0 |
| 25 | 1276987 | 21 | 20 | 59 | 1 | 4.0 |
| 26 | 1289565 | 32 | 16 | 52 | 1 | 3.0 |
| 27 | 1252564 | 34 | 13 | 53 | 0 | -6.0 |
| 28 | 1376965 | 29 | 14 | 57 | 1 | 4.0 |
| 29 | 1356345 | 35 | 12 | 53 | 1 | 3.0 |
| 30 | 1345653 | 33 | 8 | 59 | 1 | 4.0 |
| 31 | 1389567 | 27 | 12 | 61 | 0 | -8.0 |
| 32 | 1398348 | 28 | 5 | 67 | 1 | 4.0 |
| 33 | 1368947 | 32 | 5 | 63 | 1 | 6.0 |
| 34 | 1345764 | 38 | 3 | 59 | 1 | 4.0 |
| 35 | 1398064 | 36 | 1 | 63 | 1 | 5.0 |
| 36 | 1355765 | 35 | 11 | 54 | 0 | -4.0 |
| 37 | 1465736 | 35 | 12 | 53 | 1 | 6.0 |
| 38 | 1498457 | 35 | 4 | 61 | 1 | 7.0 |

In [159…

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
#fit logistic regression to the training set
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
#predict the test set results
y_pred=classifier.predict(X_test)
```

```
C:\Users\Moulya Janjarla\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A col
umn-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example
using ravel().
  return f(*args, **kwargs)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-159-494dbccd34b8> in <module>
      6 from sklearn.linear_model import LogisticRegression
      7 classifier=LogisticRegression(random_state = 0)
----> 8 classifier.fit(X_train, y_train)
      9 #predict the test set results
```

```
     10 y_pred=classifier.predict(X_test)

~\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py in fit(self, X, y, sample_weight)
   1345                                    order="C",
   1346                                    accept_large_sparse=solver != 'liblinear')
-> 1347         check_classification_targets(y)
   1348         self.classes_ = np.unique(y)
   1349

~\anaconda3\lib\site-packages\sklearn\utils\multiclass.py in check_classification_targets(y)
    181     if y_type not in ['binary', 'multiclass', 'multiclass-multioutput',
    182                       'multilabel-indicator', 'multilabel-sequences']:
--> 183         raise ValueError("Unknown label type: %r" % y_type)
    184
    185

ValueError: Unknown label type: 'continuous'
```

In [160…

```python
X = df[['GGRP','target','COS','COI','COA']]
y = df['target']
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.25,random_state=0)
```

In [161…

```python
#loading the data
x = data_scaled.iloc[:,0:9]
y = data_scaled.iloc[:,9:10]
```

In [162…

```python
clf = RandomForestClassifier(n_estimators=20)
clf.fit(X_train,y_train)
```

Out[162…

```
RandomForestClassifier(n_estimators=20)
```

In [163…

```python
y_pred=clf.predict(X_test)
```

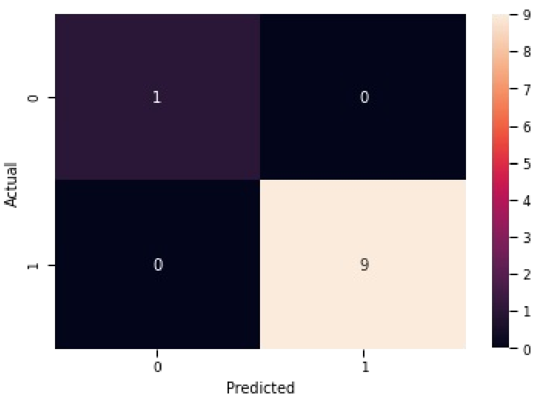
In [164…

```python
#generate confusion matrix
print('Accuracy: ', 100 * metrics.accuracy_score(y_test, y_pred))
group_names = ['True Negative','False Positive','False Negative','TruePositive']
labels = [f'{v1}' for v1 in zip(group_names)]
confusion_matrix = pd.crosstab(y_test, y_pred, rownames=['Actual'], colnames=['Predicted'])
sn.heatmap(confusion_matrix, annot=True)
```

```
Accuracy:  100.0
```

Out[164…

```
<AxesSubplot:xlabel='Predicted', ylabel='Actual'>
```




In [165…

```python
print (X_test)
```

```
     GGRP  target  COS  COI  COA
4   13.41       1   67   21   16
28  18.78       1   57   14   29
29  18.98       1   53   12   35
33  19.47       1   63    5   32
34  19.58       1   59    3   38
25  17.78       1   59   20   21
10  14.47       1   53   22   25
```

```
22  17.59       1   59   16   25
11  15.47       1   60   14   26
27  18.54       0   53   13   34
```

In [166…

```
print(y_pred)
```

```
[1 1 1 1 1 1 1 1 1 0]
```

In [167…

```
from sklearn import metrics

print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 0.0
Mean Squared Error: 0.0
Root Mean Squared Error: 0.0
```

In [168…

```
from sklearn.svm import SVC
svc = SVC(kernel = 'rbf')
svc.fit(X_train,y_train)
```

Out[168… SVC()

In [169…

```
from sklearn.metrics import confusion_matrix
y_pred_RSVM = svc.predict(X_test)
cm = confusion_matrix(y_test,y_pred_RSVM)
print('confusion matrix:\n',cm)
```

```
confusion matrix:
 [[0 1]
 [0 9]]
```

In [170…

```
from sklearn.metrics import accuracy_score
sva2 = accuracy_score(y_test,y_pred_RSVM)
print('accuracy score = ',sva2)
```

```
accuracy score =  0.9
```

In [171…

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [172…

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train,y_train)
```

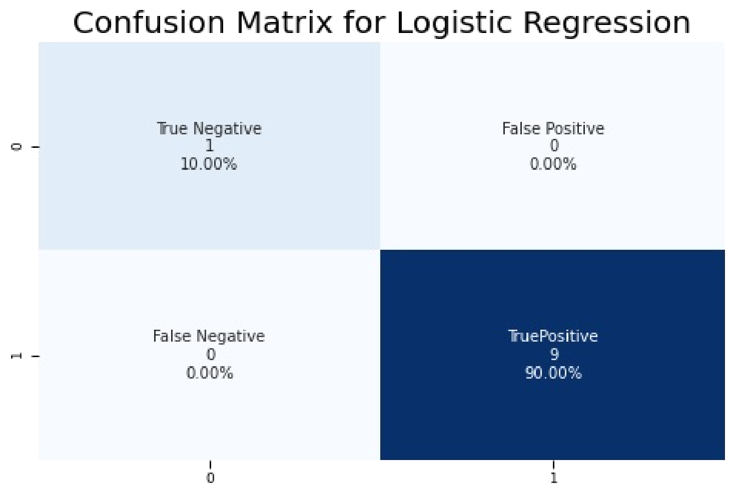
Out[172… LogisticRegression()

In [173…

```
from sklearn.metrics import confusion_matrix
y_pred_log = lr.predict(X_test)
cm = confusion_matrix(y_test,y_pred_log)
print('confusion matrix:\n',cm)
#generate confusion matrix
cm = confusion_matrix(y_test, y_pred)
group_names = ['True Negative','False Positive','False Negative','TruePositive']
group_counts = ['{0:0.0f}'.format(value) for value in cm.flatten()]
group_percentages = ['{0:.2%}'.format(value) for value in cm.flatten()/np.sum(cm)]
labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(group_names,group_counts,group_percentages)]
labels = np.asarray(labels).reshape(2,2)
sns.heatmap(cm, annot = labels, fmt = '', cmap='Blues', cbar = False)
plt.gcf().set_size_inches(8,5)
```

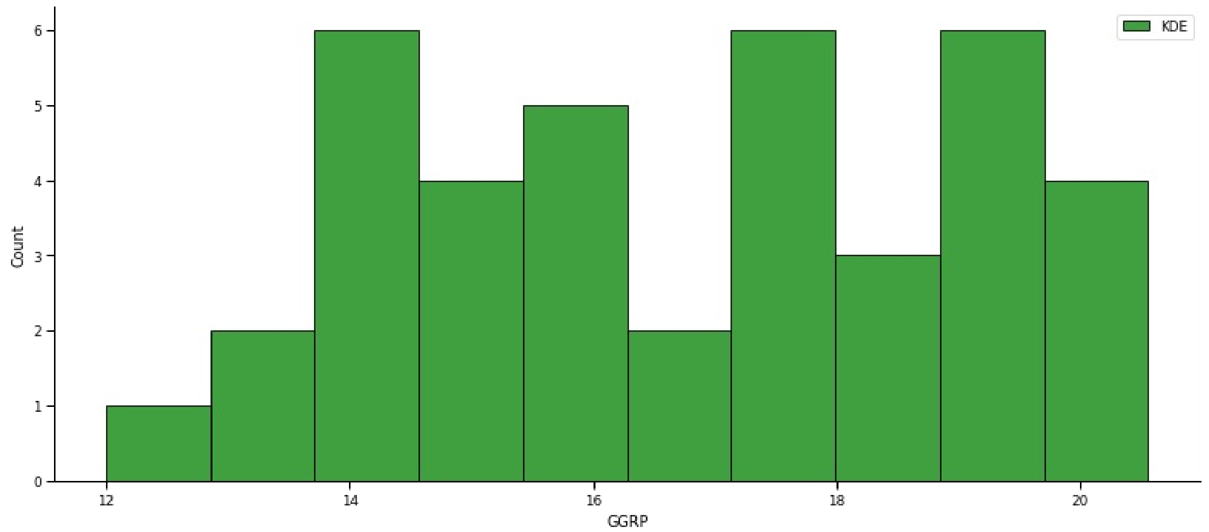
```python
plt.title('Confusion Matrix for Logistic Regression', fontsize = 20)
plt.show()
```

```
confusion matrix:
 [[1 0]
 [0 9]]
```




In [176…

```python
sns.displot(df['GGRP'],bins=10,color='green',label='KDE')
plt.legend()
plt.gcf().set_size_inches(12,5)
```




In [116…

```python
df.describe()
```

Out[116…

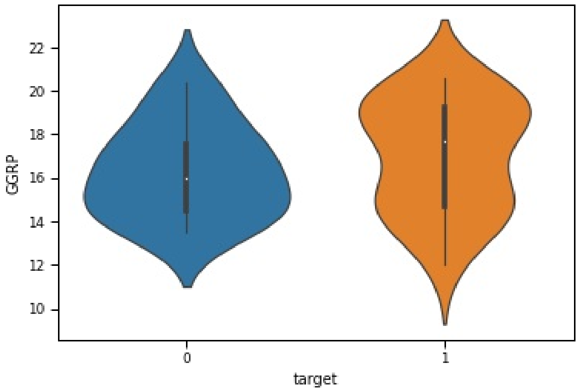
| | year | agriculture | industries | services | GGRP | increased GSDP growth(crores) | COA | COI | COS | target | incr… gr… |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 39.000000 | 39.000000 | 39.000000 | 39.000000 | 39.000000 | 3.900000e+01 | 39.000000 | 39.000000 | 39.000000 | 39.000000 | 39.000 |
| **mean** | 2031.000000 | 367199.871795 | 279920.256410 | 584165.487179 | 16.757436 | 1.125414e+06 | 27.102564 | 13.692308 | 59.282051 | 0.615385 | 0.512 |
| **std** | 11.401754 | 229687.253443 | 171150.469612 | 197488.656473 | 2.394638 | 2.560132e+05 | 6.488025 | 6.477304 | 4.650686 | 0.492864 | 4.327 |
| **min** | 2012.000000 | 76123.000000 | 92458.000000 | 209540.000000 | 12.000000 | 5.528540e+05 | 14.000000 | 1.000000 | 50.000000 | 0.000000 | -8.000 |
| **25%** | 2021.500000 | 152218.000000 | 125467.000000 | 451493.000000 | 14.630000 | 9.442325e+05 | 22.500000 | 10.000000 | 56.000000 | 0.000000 | -3.600 |
| **50%** | 2031.000000 | 325456.000000 | 222031.000000 | 598520.000000 | 16.580000 | 1.134876e+06 | 28.000000 | 14.000000 | 60.000000 | 1.000000 | 2.300 |
| **75%** | 2040.500000 | 456423.000000 | 346626.500000 | 725676.000000 | 18.880000 | 1.350764e+06 | 32.500000 | 19.000000 | 62.500000 | 1.000000 | 4.000 |
| **max** | 2050.000000 | 915236.000000 | 812547.000000 | 987215.000000 | 20.560000 | 1.520236e+06 | 38.000000 | 26.000000 | 67.000000 | 1.000000 | 7.000 |

In [177…

```python
sns.violinplot(data=df,x='target',y='GGRP')
```

Out[177…

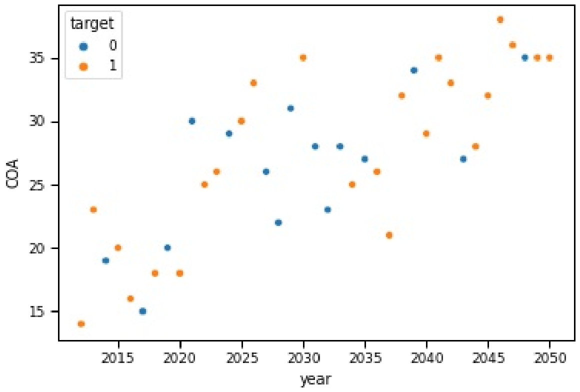
```
<AxesSubplot:xlabel='target', ylabel='GGRP'>
```




In [100…

```
sns.set_context("paper")
sns.scatterplot(x='year', y='COA',data=df, hue='target')
```

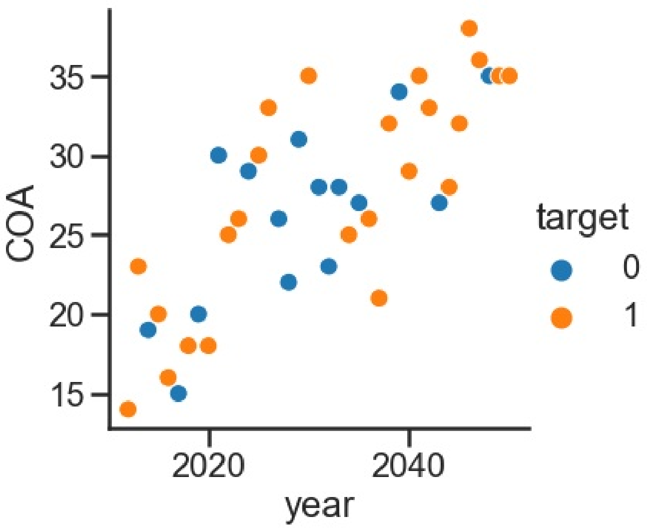
Out[100… <AxesSubplot:xlabel='year', ylabel='COA'>




In [196…

```
#Generate relation ship plot between contribution of agriculture and target
sns.set_context("poster")
sns.relplot(data=df, x='year', y='COA', hue='target')
```

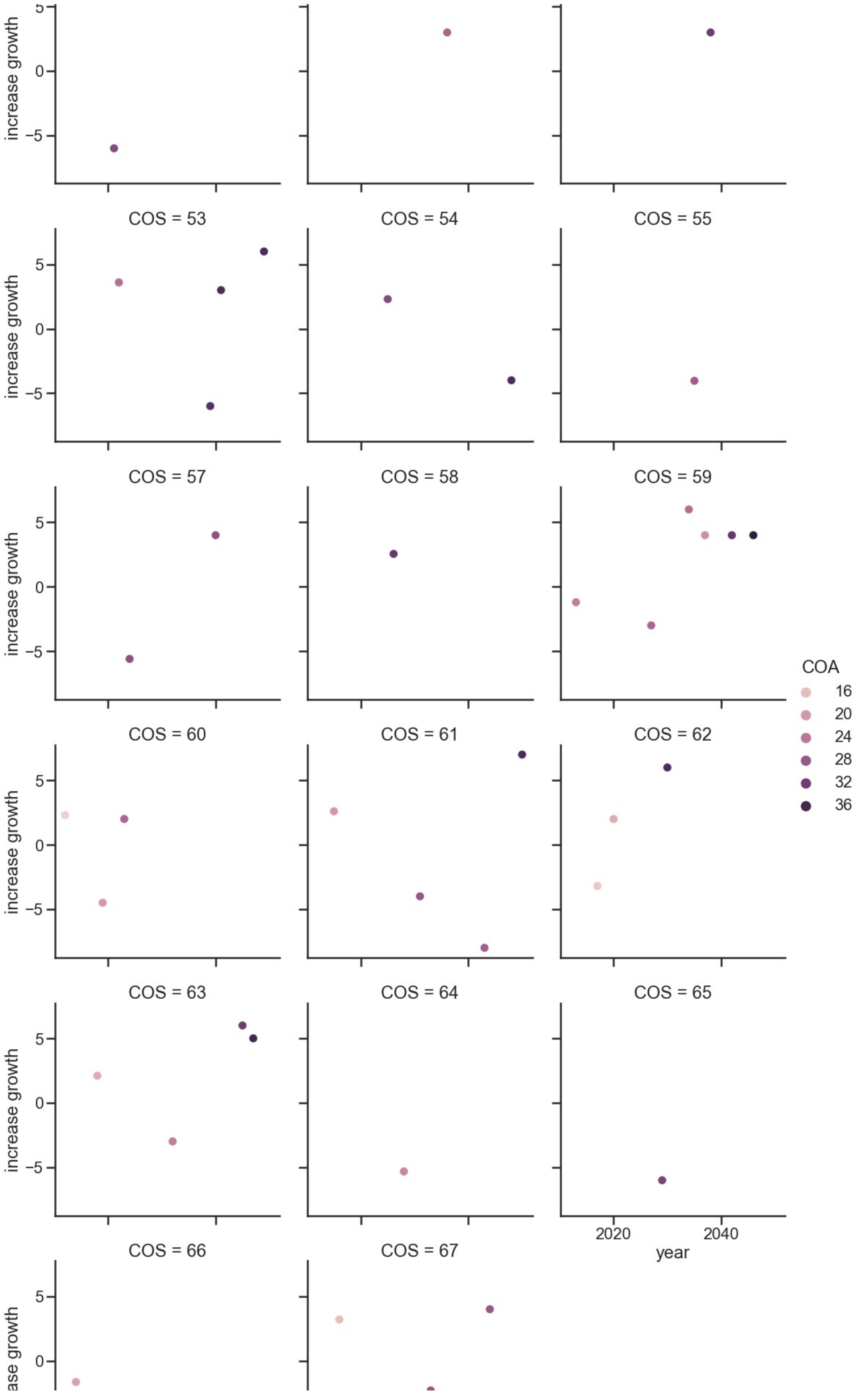
Out[196… <seaborn.axisgrid.FacetGrid at 0x1ff31d5fd90>



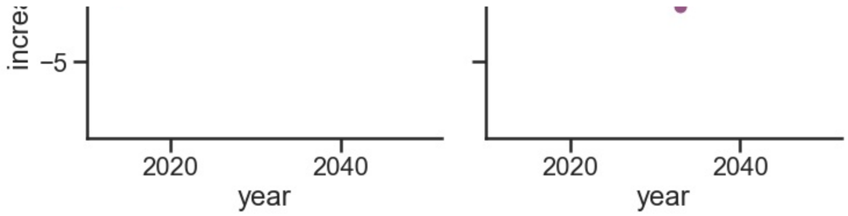

In [197…

```
sns.set_context("poster")
sns.relplot(data=df, x='year', y='increase growth',hue='COA',col='COS',col_wrap=3)
```

Out[197… <seaborn.axisgrid.FacetGrid at 0x1ff2ded3f10>



COS = 53
COS = 54
COS = 55
COS = 57
COS = 58
COS = 59
COS = 60
COS = 61
COS = 62
COS = 63
COS = 64
COS = 65
COS = 66
COS = 67
increase growth
5
0
−5
2020
2040
year
COA
16
20
24
28
32
36

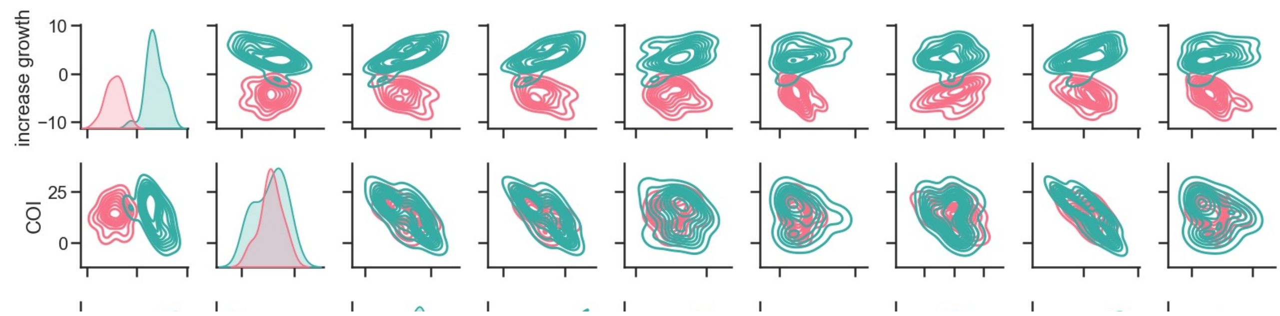


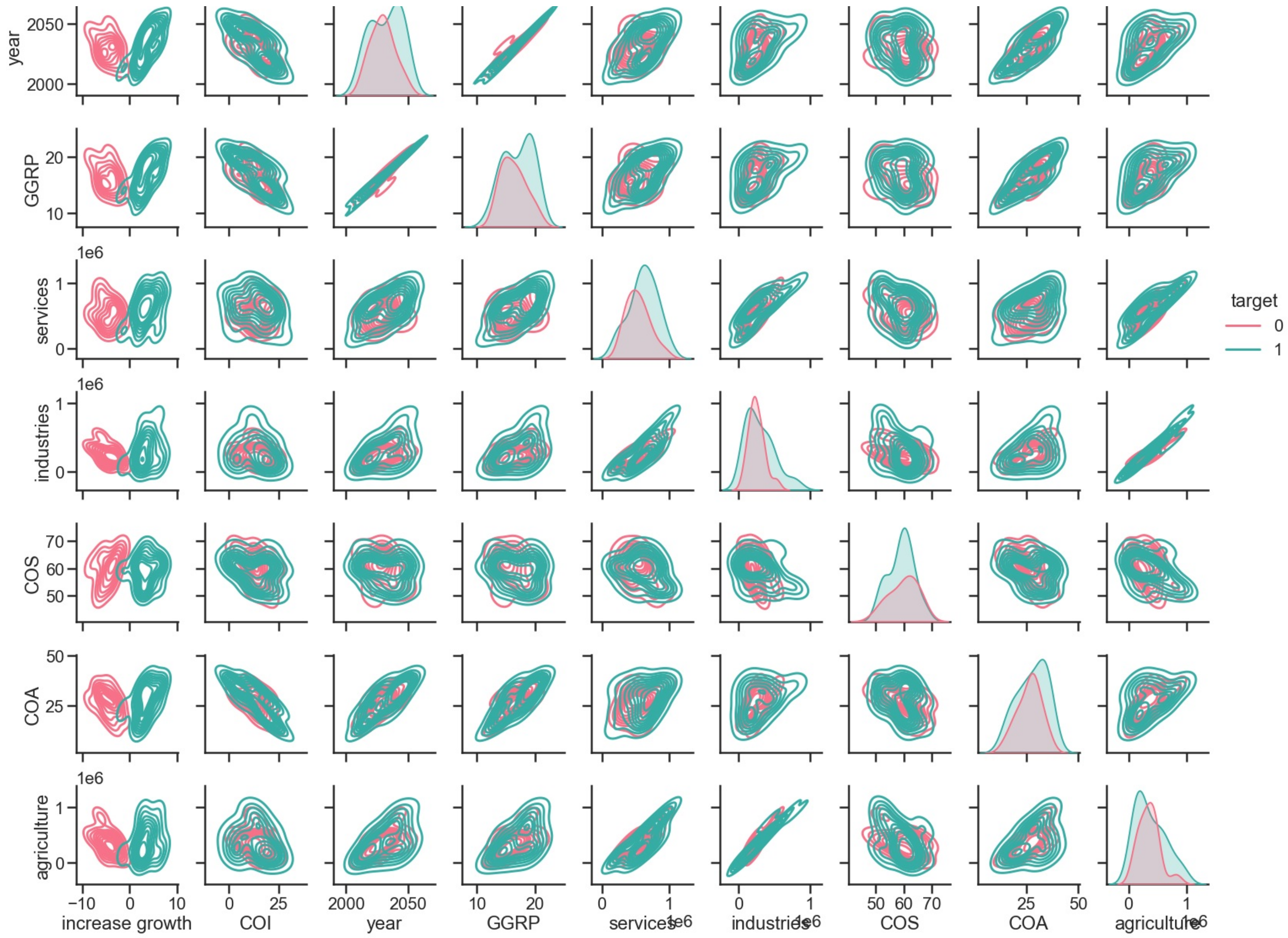

In [198…

```
sns.set_style("ticks")
sns.pairplot(df,x_vars={"year","agriculture","industries","services","GGRP","COA","COI","COS","target","increase
plt.show()
```


In [199…

```
sns.set_style("ticks")
sns.pairplot(df,x_vars={"year","agriculture","industries","services","GGRP","COA","COI","COS","increase growth"},
plt.show()
```






In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js