In [1]:
```python
import numpy as np
import pandas as pd
import statsmodels.formula.api as smf
import statsmodels.stats.api as sms
import matplotlib.pyplot as plt
```

In [2]:
```python
df = pd.read_csv('malocclusion.csv', sep = ',')
```

In [3]:
```python
df
```

Out[3]:

| | dANB | dPPPM | dIMPA | dCoA | dGoPg | dCoGo | dT | Growth | Treatment |
|---|---|---|---|---|---|---|---|---|---|
| **0** | -3.2 | -1.1 | -4.2 | 1.0 | 4.0 | 3.7 | 5 | 0 | 0 |
| **1** | -0.6 | -0.5 | 3.8 | 2.6 | -0.1 | 1.4 | 3 | 1 | 0 |
| **2** | -1.6 | -3.1 | -6.0 | 4.3 | 4.2 | 7.1 | 5 | 0 | 0 |
| **3** | -1.1 | -2.1 | -12.1 | 14.1 | 20.7 | 17.5 | 9 | 0 | 0 |
| **4** | -1.1 | 0.0 | -6.7 | 7.7 | 8.8 | 11.0 | 5 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **138** | 0.8 | -2.1 | -2.0 | 2.7 | 2.0 | 3.3 | 5 | 1 | 1 |
| **139** | 2.1 | 0.7 | 1.4 | 8.2 | 12.8 | 9.1 | 10 | 1 | 1 |
| **140** | -0.2 | -3.3 | -2.7 | 6.8 | 3.4 | 10.9 | 4 | 1 | 1 |
| **141** | 1.5 | -3.5 | 1.8 | 4.6 | 6.5 | 6.2 | 5 | 1 | 1 |
| **142** | 1.3 | -3.0 | -19.0 | 7.0 | 4.5 | 6.0 | 2 | 1 | 1 |

143 rows × 9 columns

Treatment on Growth, treatment on dANB, both ATE and ATET

1. Selection of covariates to adjust for (informed by the graph)
2. Application of most suitable adjustiment method.
3. Estimates of ATE and ATET

Naive estimator of ATE

In [4]:
```python
df.Growth[df.Treatment == 1].mean() - df.Growth[df.Treatment == 0].mean()
```

Out[4]:
```
0.1471861471861472
```

In [5]:
```python
df.dANB[df.Treatment == 1].mean() - df.dANB[df.Treatment == 0].mean()
```

Out[5]:
```
2.0287878787878784
```

The result are very biased because the data do not come from randomnized experiment - there are features we need to adjust for.

We start analyzing the graph.

# Undirected paths from Treatment to Growth.

1. Treatment <- Unobserved Cofounder -> dT -> Growth
2. Treatment <- Unobserved Cofounder -> Growth
3. Treatment -> dCoA -> dGoPg <- dT -> Growth
4. Treatment -> dCoA -> dGoPg <- dT <- Unobserved Cofounder -> Growth
5. Treatment -> dCoA -> dCoGo <- dT -> Growth
6. Treatment -> dCoA -> dCoGo <- dT <- Unobserved Cofounder -> Growth
7. Treatment -> dCoA -> dCoGo -> dPPM -> dIMPA <- dANB <- Growth

Because there is unobserved cofounder, we cannot find adjustment set that blocks all undirected path. Path2 cannot be blocked. Treatment <- Unobserved Cofounder -> Growth.

So ATE and ATET for effect of Treatment on Growth is 0.

# Directed path from Treatment to dANB

Treatment -> dANB

# Undirected paths from Treatment to dANB.

1. Treatment <- Unobserved Cofounder -> Growth -> dANB
2. Treatment <- Unobserved Cofounder -> dT -> Growth -> dANB
3. Treatment -> dCoA -> dGoPg <- dT -> Growth -> dANB
4. Treatment -> dCoA -> dGoPg <- dT <- Unobserved Cofounder -> Growth -> dANB
5. Treatment -> dCoA -> dCoGo <- dT -> Growth
6. Treatment -> dCoA -> dCoGo <- dT <- Unobserved Cofounder -> Growth -> dANB
7. Treatment -> dCoA -> dCoGo -> dPPM -> dIMPA <- dANB

Adjustment set: {Growth}. Path4, Path5, Path6 and Path 7 have collider that blocks path. Growth blocked path1 and path 2. Including collider in the adjustment set will introduce additional bias.

Let's take into Growth into account, and use linear regression to estimate ATE.

# Regression

In [6]:
```
m = smf.ols('dANB ~ Growth + Treatment', data=df)
fitted = m.fit()
print(fitted.summary())
```

```
                            OLS Regression Results
==============================================================================
=
Dep. Variable:                   dANB   R-squared:                       0.40
7
Model:                            OLS   Adj. R-squared:                  0.39
8
Method:                 Least Squares   F-statistic:                     48.0
4
Date:                Sat, 18 Sep 2021   Prob (F-statistic):           1.31e-1
6
Time:                        15:28:33   Log-Likelihood:                -251.1
7
No. Observations:                 143   AIC:                             508.
3
Df Residuals:                     140   BIC:                             517.
```

```
2
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
=
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept     -1.5600      0.181     -8.609      0.000      -1.918      -1.20
2
Growth         1.1740      0.244      4.812      0.000       0.692       1.65
6
Treatment      1.8560      0.240      7.724      0.000       1.381       2.33
1
==============================================================================
=
Omnibus:                        7.303   Durbin-Watson:                   2.12
2
Prob(Omnibus):                  0.026   Jarque-Bera (JB):                7.91
1
Skew:                           0.383   Prob(JB):                       0.019
1
Kurtosis:                       3.862   Cond. No.                         2.7
7
==============================================================================
=

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
```

ATE estimate is the coefficient of Treatment variable - 1.8560. Note that the coefficient is significantly different from $0$ as the p-value is less than $0.05$. The confidence interval contains the range of its true value.

# Propensity Score

In [7]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.calibration import CalibratedClassifierCV

# classifier to estimate the propensity score
cls = LogisticRegression()

# calibration of the classifier
cls = CalibratedClassifierCV(cls)

X = df[['Growth']]
y = df['Treatment']
cls.fit(X, y)
df['e'] = cls.predict_proba(X)[:,1].tolist()
df.head()
```

Out[7]:

| | dANB | dPPPM | dIMPA | dCoA | dGoPg | dCoGo | dT | Growth | Treatment | e |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -3.2 | -1.1 | -4.2 | 1.0 | 4.0 | 3.7 | 5 | 0 | 0 | 0.34318 |
| **1** | -0.6 | -0.5 | 3.8 | 2.6 | -0.1 | 1.4 | 3 | 1 | 0 | 0.42124 |
| **2** | -1.6 | -3.1 | -6.0 | 4.3 | 4.2 | 7.1 | 5 | 0 | 0 | 0.34318 |
| **3** | -1.1 | -2.1 | -12.1 | 14.1 | 20.7 | 17.5 | 9 | 0 | 0 | 0.34318 |
| **4** | -1.1 | 0.0 | -6.7 | 7.7 | 8.8 | 11.0 | 5 | 0 | 0 | 0.34318 |

In [8]:
```
df['w'] = df['Treatment'] / df['e'] + (1 - df['Treatment']) / (1 - df['e'])
```

In [9]:
```
m = smf.wls('dANB~ Treatment + Growth', data=df, weights=df['w'])
fitted = m.fit()
print(fitted.summary())
```

```
                            WLS Regression Results
==============================================================================
=
Dep. Variable:                   dANB   R-squared:                       0.38
6
Model:                            WLS   Adj. R-squared:                  0.37
8
Method:                 Least Squares   F-statistic:                     44.0
6
Date:                Sat, 18 Sep 2021   Prob (F-statistic):           1.44e-1
5
Time:                        15:28:34   Log-Likelihood:                -253.6
8
No. Observations:                 143   AIC:                             513.
4
Df Residuals:                     140   BIC:                             522.
3
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
=
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept     -1.5572      0.205     -7.609      0.000      -1.962      -1.15
3
Treatment      1.8570      0.241      7.701      0.000       1.380       2.33
4
Growth         1.1681      0.242      4.834      0.000       0.690       1.64
6
==============================================================================
=
Omnibus:                        4.472   Durbin-Watson:                   2.11
3
Prob(Omnibus):                  0.107   Jarque-Bera (JB):                4.08
7
Skew:                           0.312   Prob(JB):                        0.13
0
Kurtosis:                       3.545   Cond. No.                         3.1
7
==============================================================================
=

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
```

As we can see from the result, the estimated ATE - 1.8570 is very close from the result from Regression analysis.

# Matching

We can calculate the mean difference of dANB to estimate ATET by matching.

In [10]:
```
unique_on_Growth = (df.query("Treatment == 0").drop_duplicates("Growth"))
```

In [11]:
```
unique_on_Growth
```

Out[11]:

| | dANB | dPPPM | dIMPA | dCoA | dGoPg | dCoGo | dT | Growth | Treatment | e | w |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -3.2 | -1.1 | -4.2 | 1.0 | 4.0 | 3.7 | 5 | 0 | 0 | 0.34318 | 1.522486 |
| **1** | -0.6 | -0.5 | 3.8 | 2.6 | -0.1 | 1.4 | 3 | 1 | 0 | 0.42124 | 1.727833 |

In [12]:
```
matches = (df.query('Treatment == 1').merge(unique_on_Growth, on = ["Growth"]
```

In [13]:
```
matches.shape
```

Out[13]:
```
(66, 22)
```

In [14]:
```
print('Estimated ATET')
matches['t1_minus_t0'].mean()
```

```
Estimated ATET
```

Out[14]:
```
2.8045454545454547
```

We can compare our result with propensity score weighting method.

In [15]:
```
df['w1'] = df['Treatment'] + (1 - df['Treatment'])*df['e'] / (1 - df['e'])
```

In [16]:
```
m = smf.wls('dANB~ Treatment + Growth', data=df, weights=df['w1'])
fitted = m.fit()
print(fitted.summary())
```

```
                            WLS Regression Results
==============================================================================
=
Dep. Variable:                   dANB   R-squared:                       0.39
0
Model:                            WLS   Adj. R-squared:                  0.38
2
Method:                 Least Squares   F-statistic:                     44.8
1
Date:                Sat, 18 Sep 2021   Prob (F-statistic):          9.09e-1
6
Time:                        15:28:34   Log-Likelihood:                -253.2
6
No. Observations:                 143   AIC:                             512.
5
Df Residuals:                     140   BIC:                             521.
4
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
=
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept     -1.5557      0.209     -7.449      0.000      -1.969      -1.14
3
Treatment      1.8544      0.240      7.723      0.000       1.380       2.32
9
Growth         1.1683      0.237      4.931      0.000       0.700       1.63
7
==============================================================================
=
Omnibus:                        5.911   Durbin-Watson:                   2.11
8
Prob(Omnibus):                  0.052   Jarque-Bera (JB):                5.92
1
Skew:                           0.348   Prob(JB):                       0.051
8
```

```
Kurtosis:                       3.713   Cond. No.                            3.2
6
==============================================================================
=

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
```

The matching we designed above is biased. Based on propensity score weighting method, ATETis $1.8544$. We can proceed to do some sanity check using causalinference module.

In [17]:
```python
from causalinference import CausalModel
adjustment_set = ['Growth']

causal = CausalModel(
    Y=df['dANB'].values, # outcome
    D=df['Treatment'].values, # treatment
    X=df[adjustment_set].values
)
```

In [18]:
```python
causal.est_via_matching(bias_adj=True)
print(causal.estimates)
```

```
Treatment Effect Estimates: Matching

                     Est.       S.e.          z      P>|z|      [95% Conf. in
t.]
------------------------------------------------------------------------------
---
           ATE      1.856      0.237      7.829      0.000      1.392       2.
321
           ATC      1.860      0.240      7.761      0.000      1.390       2.
330
           ATT      1.852      0.240      7.723      0.000      1.382       2.
322
```

```
/home/tair/anaconda3/envs/data_science/lib/python3.8/site-packages/causalinfe
rence/estimators/matching.py:100: FutureWarning: `rcond` parameter will chang
e to the default of machine precision times ``max(M, N)`` where M and N are t
he input matrix dimensions.
To use the future default and silence this warning we advise to pass `rcond=N
one`, to keep using the old, explicitly pass `rcond=-1`.
  return np.linalg.lstsq(X, Y)[0][1:]  # don't need intercept coef
```

Indeed ATE is 1.856 and ATET/ATT is 1.852.

In [ ]: