econ434

June 8, 2024

0.0.1 Group Members:

- Hiba Farhan

- Kanupriya Parashar

[2]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import statsmodels.formula.api as smf
from linearmodels.panel import PanelOLS
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LassoCV
```

[3]:
```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

[3]:
```
file_path = 'uber_dataset.csv'
data = pd.read_csv(file_path)
data.columns
```

[3]:
```
Index(['Unnamed: 0', 'UPTTotal', 'treatUberX', 'treatGTNotStd', 'popestimate',
       'employment', 'aveFareTotal', 'VRHTotal', 'VOMSTotal', 'VRMTotal',
       'gasPrice', 'agency', 'city', 'state', 'dateSurvey'],
      dtype='object')
```

[4]:
```
data
```

[4]:
```
       Unnamed: 0  UPTTotal  treatUberX  treatGTNotStd  popestimate  \
0               0   8296756         0.0       0.000000      3163703
1               1   7847113         0.0       1.400000      3163703
2               2   9011399         0.0       3.000000      3163703
3               3   8656389         0.0       2.250000      3163703
4               4   8378406         0.0       2.600000      3163703
…             …       …         …            …          …
76208       76208     34686         1.0      14.639175       458238
```

```
76209       76209    37022        1.0     15.206185      458238
76210       76210    39197        1.0     10.309278      458238
76211       76211    31516        1.0      8.453608      458238
76212       76212    31182        1.0     12.628866      458238

       employment  aveFareTotal  VRHTotal  VOMSTotal   VRMTotal  gasPrice  \
0         1572859      0.778015  333329.0     2626.0  4740396.0     1.701
1         1581307      0.778015  310535.0     2626.0  4398939.0     1.862
2         1592152      0.778015  356761.0     2626.0  5176183.0     2.063
3         1598167      0.778015  341191.0     2626.0  4889387.0     2.121
4         1593356      0.778015  333418.0     2626.0  4747018.0     2.266
…             …            …        …         …          …         …
76208      181847           NaN    3275.0       10.0    55441.0     3.396
76209      180928           NaN    3144.0       10.0    54931.0     3.022
76210      177404           NaN    3328.0       10.0    57652.0     2.792
76211      176137           NaN    3041.0       10.0    52985.0     2.680
76212      175412           NaN    3159.0       10.0    54610.0     2.627

                                                agency     city state  \
0      King County Department of Transportation - Met…  Seattle    WA
1      King County Department of Transportation - Met…  Seattle    WA
2      King County Department of Transportation - Met…  Seattle    WA
3      King County Department of Transportation - Met…  Seattle    WA
4      King County Department of Transportation - Met…  Seattle    WA
…                                                  …        …   …
76208                                   City of Tulare  Visalia    CA
76209                                   City of Tulare  Visalia    CA
76210                                   City of Tulare  Visalia    CA
76211                                   City of Tulare  Visalia    CA
76212                                   City of Tulare  Visalia    CA

       dateSurvey
0      2004-01-01
1      2004-02-01
2      2004-03-01
3      2004-04-01
4      2004-05-01
…             …
76208  2015-08-01
76209  2015-09-01
76210  2015-10-01
76211  2015-11-01
76212  2015-12-01

[76213 rows x 15 columns]
```

1 Clean Up Dataset

[5]:
```python
# List of required columns
required_columns = ['UPTTotal', 'treatUberX', 'treatGTNotStd', 'popestimate',
                    'employment', 'aveFareTotal', 'VRHTotal', 'VOMSTotal',
                    'VRMTotal', 'gasPrice', 'dateSurvey', 'agency']

# Filter the data to only include the required columns
data = data[required_columns]

data['dateSurvey'] = pd.to_datetime(data['dateSurvey'])

data
```

```
/var/folders/qb/5tn7xwyx26ldm65g74whq05w0000gn/T/ipykernel_56828/720645514.py:9:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['dateSurvey'] = pd.to_datetime(data['dateSurvey'])
```

[5]:

	UPTTotal	treatUberX	treatGTNotStd	popestimate	employment	\
0	8296756	0.0	0.000000	3163703	1572859	
1	7847113	0.0	1.400000	3163703	1581307	
2	9011399	0.0	3.000000	3163703	1592152	
3	8656389	0.0	2.250000	3163703	1598167	
4	8378406	0.0	2.600000	3163703	1593356	
…	…	…	…	…	…	
76208	34686	1.0	14.639175	458238	181847	
76209	37022	1.0	15.206185	458238	180928	
76210	39197	1.0	10.309278	458238	177404	
76211	31516	1.0	8.453608	458238	176137	
76212	31182	1.0	12.628866	458238	175412	

	aveFareTotal	VRHTotal	VOMSTotal	VRMTotal	gasPrice	dateSurvey	\
0	0.778015	333329.0	2626.0	4740396.0	1.701	2004-01-01	
1	0.778015	310535.0	2626.0	4398939.0	1.862	2004-02-01	
2	0.778015	356761.0	2626.0	5176183.0	2.063	2004-03-01	
3	0.778015	341191.0	2626.0	4889387.0	2.121	2004-04-01	
4	0.778015	333418.0	2626.0	4747018.0	2.266	2004-05-01	
…	…	…	…	…	…	…	
76208	NaN	3275.0	10.0	55441.0	3.396	2015-08-01	
76209	NaN	3144.0	10.0	54931.0	3.022	2015-09-01	
76210	NaN	3328.0	10.0	57652.0	2.792	2015-10-01	
76211	NaN	3041.0	10.0	52985.0	2.680	2015-11-01	
76212	NaN	3159.0	10.0	54610.0	2.627	2015-12-01	

```
                                               agency
0      King County Department of Transportation - Met…
1      King County Department of Transportation - Met…
2      King County Department of Transportation - Met…
3      King County Department of Transportation - Met…
4      King County Department of Transportation - Met…
…                                                    …
76208                                 City of Tulare
76209                                 City of Tulare
76210                                 City of Tulare
76211                                 City of Tulare
76212                                 City of Tulare

[76213 rows x 12 columns]
```

[6]:
```python
data.isnull().sum()
```

[6]:
```
UPTTotal             0
treatUberX           0
treatGTNotStd    14389
popestimate          0
employment           0
aveFareTotal      4197
VRHTotal           193
VOMSTotal          147
VRMTotal           181
gasPrice             0
dateSurvey           0
agency               0
dtype: int64
```

2 if we decide to impute values

[7]:
```python
missing_columns = data.columns[data.isnull().any()]

# Impute missing values with the median using loc
for column in missing_columns:
    median_value = data[column].median()
    data.loc[:, column] = data.loc[:, column].fillna(median_value)

# Verify that there are no missing values
data.isnull().sum()
```

[7]:
```
UPTTotal         0
treatUberX       0
```

```
treatGTNotStd    0
popestimate      0
employment       0
aveFareTotal     0
VRHTotal         0
VOMSTotal        0
VRMTotal         0
gasPrice         0
dateSurvey       0
agency           0
dtype: int64
```

[8]:
```
data
```

[8]:
```
       UPTTotal  treatUberX  treatGTNotStd  popestimate  employment  \
0       8296756         0.0       0.000000      3163703     1572859
1       7847113         0.0       1.400000      3163703     1581307
2       9011399         0.0       3.000000      3163703     1592152
3       8656389         0.0       2.250000      3163703     1598167
4       8378406         0.0       2.600000      3163703     1593356
…           …         …            …          …          …
76208     34686         1.0      14.639175       458238      181847
76209     37022         1.0      15.206185       458238      180928
76210     39197         1.0      10.309278       458238      177404
76211     31516         1.0       8.453608       458238      176137
76212     31182         1.0      12.628866       458238      175412

       aveFareTotal  VRHTotal  VOMSTotal   VRMTotal  gasPrice dateSurvey  \
0          0.778015  333329.0     2626.0  4740396.0     1.701 2004-01-01
1          0.778015  310535.0     2626.0  4398939.0     1.862 2004-02-01
2          0.778015  356761.0     2626.0  5176183.0     2.063 2004-03-01
3          0.778015  341191.0     2626.0  4889387.0     2.121 2004-04-01
4          0.778015  333418.0     2626.0  4747018.0     2.266 2004-05-01
…               …         …         …          …         …        …
76208      0.914369    3275.0       10.0    55441.0     3.396 2015-08-01
76209      0.914369    3144.0       10.0    54931.0     3.022 2015-09-01
76210      0.914369    3328.0       10.0    57652.0     2.792 2015-10-01
76211      0.914369    3041.0       10.0    52985.0     2.680 2015-11-01
76212      0.914369    3159.0       10.0    54610.0     2.627 2015-12-01

                                                  agency
0      King County Department of Transportation - Met…
1      King County Department of Transportation - Met…
2      King County Department of Transportation - Met…
3      King County Department of Transportation - Met…
4      King County Department of Transportation - Met…
…                                                   …
```

```
76208                                         City of Tulare
76209                                         City of Tulare
76210                                         City of Tulare
76211                                         City of Tulare
76212                                         City of Tulare

[76213 rows x 12 columns]
```

3 Regression 1

[9]:
```python
df1 = data.copy()

# Define the dependent variable (log UPTTotal)
df1['log_UPTTotal'] = np.log(df1['UPTTotal'])

# Define the independent variables
# D_it be either treatUberX or treatGTNotStd
# W_it be the vector including remaining variables: popestimate, employment,␣
↪aveFareTotal, VRHTTotal, VOMSTotal, VRMTotal, gasPrice.
X = df1[['treatUberX', 'popestimate', 'employment', 'aveFareTotal', 'VRHTotal',␣
↪'VOMSTotal', 'VRMTotal', 'gasPrice']]
#X = sm.add_constant(X)

# Define the dependent variable
#let Yit be UPTTotal
Y = df1['log_UPTTotal']

model1 = sm.OLS(Y, X).fit()

print(model1.summary())
```

```
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:            log_UPTTotal   R-squared (uncentered):
0.943
Model:                             OLS   Adj. R-squared (uncentered):
0.943
Method:                  Least Squares   F-statistic:
1.574e+05
Date:                 Sat, 08 Jun 2024   Prob (F-statistic):
0.00
Time:                         17:00:25   Log-Likelihood:
-1.8848e+05
No. Observations:                76213   AIC:
3.770e+05
Df Residuals:                    76205   BIC:
```

```
3.771e+05
Df Model:                           8
Covariance Type:            nonrobust
================================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
treatUberX      0.1847      0.033      5.659      0.000       0.121       0.249
popestimate  -1.592e-06   4.83e-08    -32.956      0.000   -1.69e-06    -1.5e-06
employment    3.582e-06   1.04e-07     34.372      0.000    3.38e-06    3.79e-06
aveFareTotal    -0.0496      0.003    -18.720      0.000      -0.055      -0.044
VRHTotal     -1.073e-05   4.42e-07    -24.289      0.000   -1.16e-05   -9.86e-06
VOMSTotal        0.0027   6.03e-05     45.395      0.000       0.003       0.003
VRMTotal      4.913e-07   2.36e-08     20.816      0.000    4.45e-07    5.38e-07
gasPrice         3.6039      0.004    831.201      0.000       3.595       3.612
==============================================================================
Omnibus:                      140.306   Durbin-Watson:                   0.090
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              118.326
Skew:                          -0.033   Prob(JB):                     2.02e-26
Kurtosis:                       2.819   Cond. No.                     2.12e+07
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not
contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[3] The condition number is large, 2.12e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```

4 Regression 2

[10]:

```python
df2 = data.copy()

df2['log_UPTTotal'] = np.log(df2['UPTTotal'])

# Set the index for panel data
# agency and dateSurvey are included to add fixed effects for the entity and␣
↪time.
df2 = df2.set_index(['agency', 'dateSurvey'])

# EntityEffects and TimeEffects are included to add fixed effects for the␣
↪entity and time.
formula = 'log_UPTTotal ~ treatUberX + popestimate + employment + aveFareTotal␣
↪+ VRHTotal + VOMSTotal + VRMTotal + gasPrice + EntityEffects + TimeEffects'

model2 = PanelOLS.from_formula(formula, data=df2).fit()
```

```python
print(model2)
```

```
                          PanelOLS Estimation Summary
================================================================================
Dep. Variable:            log_UPTTotal   R-squared:                        0.0220
Estimator:                    PanelOLS   R-squared (Between):              0.1085
No. Observations:                76213   R-squared (Within):               0.0208
Date:                 Sat, Jun 08 2024   R-squared (Overall):              0.1078
Time:                         17:00:29   Log-likelihood                 -2.257e+04
Cov. Estimator:             Unadjusted
                                         F-statistic:                      212.23
Entities:                          703   P-value                           0.0000
Avg Obs:                        108.41   Distribution:                 F(8,75359)
Min Obs:                        5.0000
Max Obs:                        222.00   F-statistic (robust):             212.23
                                         P-value                           0.0000
Time periods:                      144   Distribution:                 F(8,75359)
Avg Obs:                        529.26
Min Obs:                        387.00
Max Obs:                        612.00

                              Parameter Estimates
================================================================================
              Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
--------------------------------------------------------------------------------
treatUberX      -0.0606     0.0062    -9.8442     0.0000     -0.0727     -0.0486
popestimate    6.54e-08  1.292e-08     5.0624     0.0000   4.008e-08   9.072e-08
employment    2.632e-07  2.166e-08     12.148     0.0000   2.207e-07   3.056e-07
aveFareTotal    -0.0032     0.0005    -7.0332     0.0000     -0.0041     -0.0023
VRHTotal      8.495e-07   1.65e-07     5.1497     0.0000   5.262e-07   1.173e-06
VOMSTotal        0.0004    2.2e-05     19.186     0.0000      0.0004      0.0005
VRMTotal     -2.508e-08  7.398e-09    -3.3901     0.0007  -3.958e-08  -1.058e-08
gasPrice        -0.0137     0.0156    -0.8799     0.3789     -0.0443      0.0168
================================================================================

F-test for Poolability: 2045.1
P-value: 0.0000
Distribution: F(845,75359)

Included effects: Entity, Time
```

5 Regression 3

[11]:
```python
df3 = data.copy()

df3['log_UPTTotal'] = np.log(df3['UPTTotal'])

# Create the dummy variable P_it
median_population = df3['popestimate'].median()
df3['P_it'] = (df3['popestimate'] > median_population).astype(int)

df3 = df3.set_index(['agency', 'dateSurvey'])

formula = 'log_UPTTotal ~ treatUberX + treatUberX * P_it + popestimate +␣
↪employment + aveFareTotal + VRHTotal + VOMSTotal + VRMTotal + gasPrice +␣
↪EntityEffects + TimeEffects'

model3 = PanelOLS.from_formula(formula, data=df3).fit()


print(model3)
```

```
                          PanelOLS Estimation Summary
================================================================================
Dep. Variable:            log_UPTTotal   R-squared:                        0.0223
Estimator:                    PanelOLS   R-squared (Between):              0.1136
No. Observations:                76213   R-squared (Within):               0.0245
Date:                 Sat, Jun 08 2024   R-squared (Overall):              0.1128
Time:                         17:00:33   Log-likelihood                 -2.256e+04
Cov. Estimator:             Unadjusted
                                         F-statistic:                      171.63
Entities:                          703   P-value                           0.0000
Avg Obs:                        108.41   Distribution:                F(10,75357)
Min Obs:                        5.0000
Max Obs:                        222.00   F-statistic (robust):             171.63
                                         P-value                           0.0000
Time periods:                      144   Distribution:                F(10,75357)
Avg Obs:                        529.26
Min Obs:                        387.00
Max Obs:                        612.00

                              Parameter Estimates
================================================================================
===
                 Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper
CI
--------------------------------------------------------------------------------
---
treatUberX         -0.0303     0.0094    -3.2209     0.0013     -0.0488
```

```
-0.0119
P_it                0.0028     0.0199     0.1413     0.8876     -0.0361
0.0417
popestimate      6.607e-08   1.32e-08     5.0049     0.0000   4.019e-08
9.194e-08
employment       2.714e-07  2.177e-08     12.468     0.0000   2.288e-07
3.141e-07
aveFareTotal       -0.0033     0.0005    -7.0818     0.0000     -0.0042
-0.0024
VRHTotal         8.483e-07  1.649e-07     5.1427     0.0000    5.25e-07
1.172e-06
VOMSTotal           0.0004  2.203e-05     19.329     0.0000      0.0004
0.0005
VRMTotal        -2.516e-08  7.397e-09    -3.4015     0.0007  -3.966e-08
-1.066e-08
gasPrice           -0.0083     0.0156    -0.5303     0.5959     -0.0389
0.0224
treatUberX:P_it    -0.0397     0.0094    -4.2414     0.0000     -0.0580
-0.0214
===============================================================================
===

F-test for Poolability: 1998.6
P-value: 0.0000
Distribution: F(845,75357)

Included effects: Entity, Time
```

6 Regression 4

[12]:

```python
df4 = data.copy()

df4['log_UPTTotal'] = np.log(df4['UPTTotal'])

# Create the dummy variable F_it
median_rides = df4['UPTTotal'].median()
df4['F_it'] = (df4['UPTTotal'] > median_rides).astype(int)

df4 = df4.set_index(['agency', 'dateSurvey'])

formula = 'log_UPTTotal ~ treatUberX + treatUberX * F_it + popestimate +␣
↪employment + aveFareTotal + VRHTotal + VOMSTotal + VRMTotal + gasPrice +␣
↪EntityEffects + TimeEffects'

model4 = PanelOLS.from_formula(formula, data=df4).fit()

print(model4)
```

```
                          PanelOLS Estimation Summary
================================================================================
Dep. Variable:            log_UPTTotal   R-squared:                        0.1155
Estimator:                    PanelOLS   R-squared (Between):              0.1292
No. Observations:                76213   R-squared (Within):               0.1335
Date:                 Sat, Jun 08 2024   R-squared (Overall):              0.1356
Time:                         17:00:36   Log-likelihood                 -1.874e+04
Cov. Estimator:             Unadjusted
                                         F-statistic:                      984.09
Entities:                          703   P-value                           0.0000
Avg Obs:                        108.41   Distribution:                F(10,75357)
Min Obs:                        5.0000
Max Obs:                        222.00   F-statistic (robust):             984.09
                                         P-value                           0.0000
Time periods:                      144   Distribution:                F(10,75357)
Avg Obs:                        529.26
Min Obs:                        387.00
Max Obs:                        612.00

                             Parameter Estimates
================================================================================
===
                 Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper
CI
--------------------------------------------------------------------------------
---
treatUberX          0.0092     0.0076     1.2126     0.2253     -0.0057
0.0241
F_it                0.6132     0.0069     89.228     0.0000      0.5997
0.6266
```

```
popestimate      1.221e-08   1.23e-08     0.9928     0.3208   -1.19e-08
3.633e-08
employment       2.326e-07  2.061e-08     11.288     0.0000   1.922e-07
2.73e-07
aveFareTotal       -0.0030     0.0004    -6.7820     0.0000     -0.0038
-0.0021
VRHTotal           9.38e-07  1.569e-07     5.9790     0.0000   6.305e-07
1.246e-06
VOMSTotal            0.0004  2.096e-05     19.401     0.0000      0.0004
0.0004
VRMTotal        -3.016e-08  7.036e-09    -4.2869     0.0000  -4.396e-08
-1.637e-08
gasPrice             0.0032     0.0148     0.2186     0.8269     -0.0258
0.0323
treatUberX:F_it    -0.0974     0.0077    -12.623     0.0000     -0.1125
-0.0823
================================================================================
===

F-test for Poolability: 853.69
P-value: 0.0000
Distribution: F(845,75357)

Included effects: Entity, Time
```

7 Regression 5

[24]:
```python
df5 = data.copy()

df5['log_UPTTotal'] = np.log(df5['UPTTotal'])

# Create the dummy variable P_it
median_population = df5['popestimate'].median()
df5['P_it'] = (df5['popestimate'] > median_population).astype(int)

# Create interaction term D_it * P_it
df5['D_it_P_it'] = df5['treatUberX'] * df5['P_it']



# Create entity and time dummies
df5 = pd.get_dummies(df5, columns=['agency', 'dateSurvey'], drop_first=True)

y = df5['log_UPTTotal']



variables = [
    'treatUberX', 'popestimate', 'employment', 'aveFareTotal',
```

```
    'VRHTotal', 'VOMSTotal', 'VRMTotal', 'gasPrice', 'D_it_P_it'
] + [col for col in df5.columns if col.startswith('agency_') or col.
 ↪startswith('dateSurvey_')]

X = df5[variables]

# Scale the independent variables
scaler = StandardScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)

# Fit the LASSO model with cross-validation
model5 = LassoCV(cv=10)
model5.fit(X_scaled, y)

# Print the coefficients with column names, excluding dummies
coef = pd.Series(model5.coef_, index=X.columns)
main_variables = [col for col in X.columns if not (col.startswith('agency_') or␣
 ↪col.startswith('dateSurvey_'))]
print("LASSO Coefficients for main variables:\n", coef[main_variables])

# Print the R^2 score
r2_score = model5.score(X_scaled, y)
print(f"R^2: {r2_score:.4f}")

# Print the best alpha value
print(f"Best alpha: {model5.alpha_}")
```

```
LASSO Coefficients for main variables:
 treatUberX      0.000000
popestimate     0.000000
employment      0.000000
aveFareTotal   -0.000000
VRHTotal        0.000000
VOMSTotal       0.578753
VRMTotal        0.000000
gasPrice        0.000000
D_it_P_it       0.000000
dtype: float64
R^2: 0.2265
Best alpha: 0.4417525406804395
```

8 Regression 6

[25]:
```
df6 = data.copy()

df6['log_UPTTotal'] = np.log(df6['UPTTotal'])
```

```python
# Create the dummy variable F_it
median_rides = df6['UPTTotal'].median()
df6['F_it'] = (df6['UPTTotal'] > median_rides).astype(int)

# Create interaction term D_it * F_it
df6['D_it_F_it'] = df6['treatUberX'] * df6['F_it']

# Create entity and time dummies
df6 = pd.get_dummies(df6, columns=['agency', 'dateSurvey'], drop_first=True)

y = df6['log_UPTTotal']

variables = [
    'treatUberX', 'popestimate', 'employment', 'aveFareTotal',
    'VRHTotal', 'VOMSTotal', 'VRMTotal', 'gasPrice', 'D_it_F_it'
] + [col for col in df5.columns if col.startswith('agency_') or col.
 ↪startswith('dateSurvey_')]

X = df6[variables]

#X = df6.drop(columns=['UPTTotal', 'log_UPTTotal'])

# Scale the independent variables
scaler = StandardScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)

# Fit the LASSO model with cross-validation and increased iterations
model6 = LassoCV(cv=10, max_iter=10000)
model6.fit(X_scaled, y)

# Print the coefficients with column names, excluding dummies
coef = pd.Series(model6.coef_, index=X.columns)
main_variables = [col for col in X.columns if not (col.startswith('agency_') or␣
 ↪col.startswith('dateSurvey_'))]
print("LASSO Coefficients for main variables:\n", coef[main_variables])

# Print the R^2 score
r2_score = model6.score(X_scaled, y)
print(f"R^2: {r2_score:.4f}")

# Print the best alpha value
print(f"Best alpha: {model6.alpha_}")
```

```
LASSO Coefficients for main variables:
 treatUberX      0.000000
popestimate     0.000000
employment      0.000000
```

```
aveFareTotal   -0.000000
VRHTotal        0.000000
VOMSTotal       0.655866
VRMTotal        0.000000
gasPrice        0.000000
D_it_F_it       0.048678
dtype: float64
R^2: 0.2531
Best alpha: 0.3583191053255799
```

9 Regression 7

[26]:
```
dt = data.copy()
dt['year_month'] = dt['dateSurvey'].dt.to_period('M')
dt.head()
```

[26]:
```
    UPTTotal  treatUberX  treatGTNotStd  popestimate  employment  aveFareTotal  \
0    8296756         0.0           0.00      3163703     1572859      0.778015
1    7847113         0.0           1.40      3163703     1581307      0.778015
2    9011399         0.0           3.00      3163703     1592152      0.778015
3    8656389         0.0           2.25      3163703     1598167      0.778015
4    8378406         0.0           2.60      3163703     1593356      0.778015

   VRHTotal  VOMSTotal   VRMTotal  gasPrice dateSurvey  \
0  333329.0     2626.0  4740396.0     1.701 2004-01-01
1  310535.0     2626.0  4398939.0     1.862 2004-02-01
2  356761.0     2626.0  5176183.0     2.063 2004-03-01
3  341191.0     2626.0  4889387.0     2.121 2004-04-01
4  333418.0     2626.0  4747018.0     2.266 2004-05-01

                                            agency year_month
0  King County Department of Transportation - Met…    2004-01
1  King County Department of Transportation - Met…    2004-02
2  King County Department of Transportation - Met…    2004-03
3  King County Department of Transportation - Met…    2004-04
4  King County Department of Transportation - Met…    2004-05
```

[27]:
```
dt = pd.get_dummies(dt, columns=['year_month', 'agency'], drop_first=True)
```

[28]:
```
dt.head()
```

[28]:
```
    UPTTotal  treatUberX  treatGTNotStd  popestimate  employment  aveFareTotal  \
0    8296756         0.0           0.00      3163703     1572859      0.778015
1    7847113         0.0           1.40      3163703     1581307      0.778015
2    9011399         0.0           3.00      3163703     1592152      0.778015
3    8656389         0.0           2.25      3163703     1598167      0.778015
4    8378406         0.0           2.60      3163703     1593356      0.778015
```

```
   VRHTotal  VOMSTotal   VRMTotal  gasPrice  …  \
0  333329.0     2626.0  4740396.0     1.701  …
1  310535.0     2626.0  4398939.0     1.862  …
2  356761.0     2626.0  5176183.0     2.063  …
3  341191.0     2626.0  4889387.0     2.121  …
4  333418.0     2626.0  4747018.0     2.266  …

  agency_Yolo County Transportation District  \
0                                      False
1                                      False
2                                      False
3                                      False
4                                      False

   agency_York County Transportation Authority  \
0                                        False
1                                        False
2                                        False
3                                        False
4                                        False

   agency_Yuba-Sutter Transit Authority  \
0                                 False
1                                 False
2                                 False
3                                 False
4                                 False

   agency_Yuma County Intergovernmental Public Transportation Authority  \
0                                              False
1                                              False
2                                              False
3                                              False
4                                              False

   agency_Yuma Metropolitan Planning Organization  \
0                                           False
1                                           False
2                                           False
3                                           False
4                                           False

   agency_vRide, Inc. - Anchorage  agency_vRide, Inc. - Atlanta  \
0                           False                         False
1                           False                         False
2                           False                         False
```

```
3                          False                          False
4                          False                          False

   agency_vRide, Inc. - Denver  agency_vRide, Inc. - Tucson  \
0                       False                        False
1                       False                        False
2                       False                        False
3                       False                        False
4                       False                        False

   agency_vRide, Inc. - Valley Metro
0                             False
1                             False
2                             False
3                             False
4                             False

[5 rows x 856 columns]
```

[29]:
```
dt.set_index(['dateSurvey'], inplace=True)
```

[30]:
```
y = np.log(dt["UPTTotal"])
d = dt["treatUberX"].values
dt1 = pd.DataFrame(dt)
w_cols = ["popestimate","employment", "aveFareTotal", "VRHTotal", "VOMSTotal",␣
 ↪"VRMTotal", "gasPrice"]
w = dt1[w_cols].values
```

[31]:
```
#Creating dummy
median_population = dt['popestimate'].median()
dt['P_it'] = (dt['popestimate'] > median_population).astype(int)
```

[32]:
```
d_p = (d * dt['P_it'])
dt["d_p"] = d_p
d_p = dt["d_p"].values
fixed_effect_cols = [col for col in dt1.columns if col.
 ↪startswith('year_month_') or col.startswith('agency_')]
fixed_effects = dt1[fixed_effect_cols].values
print("Shape of d:", d.shape)
print("Shape of d_p:", d_p.shape)
print("Shape of w:", w.shape)
print("Shape of fixed effects:", fixed_effects.shape)
```

```
Shape of d: (76213,)
Shape of d_p: (76213,)
Shape of w: (76213, 7)
Shape of fixed effects: (76213, 845)
```

[33]:
```python
x_dash = np.column_stack((d, d_p, w, fixed_effects))
```

[34]:
```python
import numpy as np
from sklearn.linear_model import LassoCV, LinearRegression
from statsmodels.api import add_constant, OLS
```

[35]:
```python
max_iterations = 10000 # I ADDED THIS, AND ADDEDD TO YOUR LASSO_Y FORMULAA TOO
#first LASSO
lasso_y = LassoCV(cv=5, max_iter=max_iterations).fit(x_dash,y)
```

[36]:
```python
coefficients = lasso_y.coef_
feature_names = ['d', 'd_p'] + w_cols + fixed_effect_cols
coef_df = pd.DataFrame({'Feature': feature_names, 'Coefficient': coefficients})
```

[37]:
```python
coef_df.head()
```

[37]:

	Feature	Coefficient
0	d	0.000000e+00
1	d_p	0.000000e+00
2	popestimate	1.272980e-08
3	employment	0.000000e+00
4	aveFareTotal	-0.000000e+00

[38]:
```python
selected_features_y = np.where(lasso_y.coef_ != 0)[0]
```

[39]:
```python
#second LASSO
x_dash1 = np.column_stack((w, fixed_effects))
lasso_d = LassoCV(cv=5, max_iter=max_iterations).fit(x_dash1,d)
selected_features_d = np.where(lasso_d.coef_ != 0)[0]
```

[40]:
```python
#second LASSO
lasso_dp= LassoCV(cv=10, max_iter=max_iterations).fit(x_dash1,d_p)
```

[41]:
```python
#OLS of y on selected covariates
selected_features_dp = np.where(lasso_dp.coef_ != 0)[0]
selected_features = np.union1d(np.union1d(selected_features_y,␣
 ↪selected_features_d), selected_features_dp)
Z_selected = x_dash[:, selected_features]
X_final = np.column_stack((d,d_p, Z_selected))
ols_model = sm.OLS(y, X_final).fit()
ols_model.summary()
```

[41]:

Dep. Variable:	UPTTotal	**R-squared (uncentered):**	0.372
Model:	OLS	**Adj. R-squared (uncentered):**	0.372
Method:	Least Squares	**F-statistic:**	9045.
Date:	Sat, 08 Jun 2024	**Prob (F-statistic):**	0.00
Time:	16:33:58	**Log-Likelihood:**	-2.7986e+05
No. Observations:	76213	**AIC:**	5.597e+05
Df Residuals:	76208	**BIC:**	5.598e+05
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
x1	5.6974	0.103	55.429	0.000	5.496	5.899
x2	-3.5568	0.119	-29.839	0.000	-3.790	-3.323
x3	5.6974	0.103	55.429	0.000	5.496	5.899
x4	-3.5568	0.119	-29.839	0.000	-3.790	-3.323
x5	9.267e-07	6.61e-09	140.147	0.000	9.14e-07	9.4e-07
x6	1.008e-05	1.11e-06	9.098	0.000	7.91e-06	1.22e-05
x7	5.266e-08	7.62e-08	0.691	0.489	-9.67e-08	2.02e-07

Omnibus:	32462.408	**Durbin-Watson:**	0.009
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	159257.829
Skew:	-2.056	**Prob(JB):**	0.00
Kurtosis:	8.765	**Cond. No.**	1.73e+19

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The smallest eigenvalue is 9.51e-21. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

[42]:
```python
#After Double-LASSO variables and coefficients
treatment_vars = ['d', 'd_p']
fixed_effect_names = fixed_effect_cols
all_variable_names = treatment_vars + w_cols + fixed_effect_names
coefficients7 = ols_model.params
selected_variable_names7 = []
for idx in selected_features:
    selected_variable_names7.append(all_variable_names[idx])
selected_variable_names7 = ['d', 'd_p'] + selected_variable_names7
result7 = pd.DataFrame({
    'Feature': selected_variable_names7,
    'Coefficient': coefficients7
})

result7
```

[42]:

	Feature	Coefficient
x1	d	5.697352e+00
x2	d_p	-3.556761e+00

```
x3            d  5.697352e+00
x4          d_p -3.556761e+00
x5  popestimate  9.267066e-07
x6     VRHTotal  1.007846e-05
x7     VRMTotal  5.266218e-08
```

10 Regression 8

[43]:
```
# Creating the dummy variable F_it
median_rides = dt['UPTTotal'].median()
dt['F_it'] = (dt['UPTTotal'] > median_rides).astype(int)
```

[44]:
```
d_f = (d * dt['F_it'])
dt["d_f"]=d_f
d_f = dt["d_f"].values
```

[45]:
```
x_dash2 = np.column_stack((d, d_f, w, fixed_effects))
```

[46]:
```
#first LASSO
lasso_y1 = LassoCV(cv=10, max_iter=max_iterations).fit(x_dash2,y)
```

[47]:
```
coefficients1 = lasso_y1.coef_
feature_names1 = ['d', 'd_f'] + w_cols + fixed_effect_cols
coef_df1 = pd.DataFrame({'Feature': feature_names1, 'Coefficient':␣
 ↪coefficients1})
```

[48]:
```
coef_df1.head()
```

[48]:

	Feature	Coefficient
0	d	0.000000e+00
1	d_f	0.000000e+00
2	popestimate	1.272980e-08
3	employment	0.000000e+00
4	aveFareTotal	-0.000000e+00

[49]:
```
selected_features_y1 = np.where(lasso_y1.coef_ != 0)[0]
```

[50]:
```
#second Lasso
lasso_d1 = LassoCV(cv=10, max_iter=max_iterations).fit(x_dash1,d)
```

[51]:
```
selected_features_d1 = np.where(lasso_d1.coef_ != 0)[0]
```

[52]:
```
#second LASSO
lasso_df = LassoCV(cv=10, max_iter=max_iterations).fit(x_dash1,d_f)
selected_features_df = np.where(lasso_df.coef_ != 0)[0]
```

[53]:
```
#OLS of y on selected variables
selected_features1 = np.union1d(np.union1d(selected_features_y1,␣
↪selected_features_d1), selected_features_df)
Z_selected1 = x_dash2[:, selected_features1]
X_final1 = np.column_stack((d,d_f, Z_selected1))
ols_model1 = sm.OLS(y, X_final1).fit()
ols_model1.summary()
```

[53]:

Dep. Variable:	UPTTotal	**R-squared (uncentered):**	0.366
Model:	OLS	**Adj. R-squared (uncentered):**	0.366
Method:	Least Squares	**F-statistic:**	8795.
Date:	Sat, 08 Jun 2024	**Prob (F-statistic):**	0.00
Time:	16:39:11	**Log-Likelihood:**	-2.8025e+05
No. Observations:	76213	**AIC:**	5.605e+05
Df Residuals:	76208	**BIC:**	5.606e+05
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
x1	2.4592	0.081	30.504	0.000	2.301	2.617
x2	1.0002	0.101	9.887	0.000	0.802	1.198
x3	2.4592	0.081	30.504	0.000	2.301	2.617
x4	1.0002	0.101	9.887	0.000	0.802	1.198
x5	8.913e-07	6.53e-09	136.474	0.000	8.78e-07	9.04e-07
x6	9.789e-06	1.11e-06	8.792	0.000	7.61e-06	1.2e-05
x7	5.843e-08	7.66e-08	0.763	0.446	-9.17e-08	2.09e-07

Omnibus:	34221.541	**Durbin-Watson:**	0.007
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	184256.552
Skew:	-2.149	**Prob(JB):**	0.00
Kurtosis:	9.289	**Cond. No.**	1.35e+19

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The smallest eigenvalue is 1.57e-20. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

[54]:
```
#After Double-LASSO variables and coefficients
treatment_vars = ['d', 'd_f']
fixed_effect_names = fixed_effect_cols
all_variable_names = treatment_vars + w_cols + fixed_effect_names
coefficients8 = ols_model1.params
selected_variable_names8 = []
for idx in selected_features1:
    selected_variable_names8.append(all_variable_names[idx])
selected_variable_names8 = ['d', 'd_f'] + selected_variable_names8
result8 = pd.DataFrame({
    'Feature': selected_variable_names8,
```

```
    'Coefficient': coefficients8
})

result8
```

[54]:

```
        Feature   Coefficient
x1            d  2.459209e+00
x2          d_f  1.000206e+00
x3            d  2.459209e+00
x4          d_f  1.000206e+00
x5  popestimate  8.912887e-07
x6     VRHTotal  9.788938e-06
x7     VRMTotal  5.842665e-08
```

11 Regression 9

[55]:

```
from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree=5, interaction_only=True, include_bias=False)
w_tilde = poly.fit_transform(w)

feature_names_dash = poly.get_feature_names_out(w_cols)

w_tilde_df = pd.DataFrame(w_tilde, columns=feature_names_dash)

w_tilde_df.head()
```

[55]:

```
   popestimate  employment  aveFareTotal  VRHTotal  VOMSTotal   VRMTotal  \
0    3163703.0   1572859.0      0.778015  333329.0     2626.0  4740396.0
1    3163703.0   1581307.0      0.778015  310535.0     2626.0  4398939.0
2    3163703.0   1592152.0      0.778015  356761.0     2626.0  5176183.0
3    3163703.0   1598167.0      0.778015  341191.0     2626.0  4889387.0
4    3163703.0   1593356.0      0.778015  333418.0     2626.0  4747018.0

   gasPrice  popestimate employment  popestimate aveFareTotal  \
0     1.701              4.976059e+12              2.461408e+06
1     1.862              5.002786e+12              2.461408e+06
2     2.063              5.037096e+12              2.461408e+06
3     2.121              5.056126e+12              2.461408e+06
4     2.266              5.040905e+12              2.461408e+06

   popestimate VRHTotal  …  \
0          1.054554e+12  …
1          9.824405e+11  …
2          1.128686e+12  …
3          1.079427e+12  …
```

```
4         1.054836e+12  …

   popestimate aveFareTotal VRHTotal VOMSTotal gasPrice  \
0                                       3.664846e+15
1                                       3.737392e+15
2                                       4.757239e+15
3                                       4.677530e+15
4                                       4.883457e+15

   popestimate aveFareTotal VRHTotal VRMTotal gasPrice  \
0                                       6.615698e+18
1                                       6.260685e+18
2                                       9.377129e+18
3                                       8.709161e+18
4                                       8.827820e+18

   popestimate aveFareTotal VOMSTotal VRMTotal gasPrice  \
0                                       5.211915e+16
1                                       5.294269e+16
2                                       6.902195e+16
3                                       6.703065e+16
4                                       6.952791e+16

   popestimate VRHTotal VOMSTotal VRMTotal gasPrice  \
0                                      2.232968e+22
1                                      2.113142e+22
2                                      3.165021e+22
3                                      2.939565e+22
4                                      2.979616e+22

   employment aveFareTotal VRHTotal VOMSTotal VRMTotal  \
0                                       5.077620e+21
1                                       4.413238e+21
2                                       6.006952e+21
3                                       5.446993e+21
4                                       5.152351e+21

   employment aveFareTotal VRHTotal VOMSTotal gasPrice  \
0                                       1.822006e+15
1                                       1.868053e+15
2                                       2.394108e+15
3                                       2.362888e+15
4                                       2.459486e+15

   employment aveFareTotal VRHTotal VRMTotal gasPrice  \
0                                       3.289045e+18
1                                       3.129265e+18
```

```
2                                       4.719095e+18
3                                       4.399494e+18
4                                       4.446012e+18

   employment aveFareTotal VOMSTotal VRMTotal gasPrice  \
0                                       2.591143e+16
1                                       2.646223e+16
2                                       3.473570e+16
3                                       3.386101e+16
4                                       3.501679e+16

   employment VRHTotal VOMSTotal VRMTotal gasPrice  \
0                                     1.110137e+22
1                                     1.056207e+22
2                                     1.592815e+22
3                                     1.484942e+22
4                                     1.500643e+22

   aveFareTotal VRHTotal VOMSTotal VRMTotal gasPrice
0                                       5.491294e+15
1                                       5.196619e+15
2                                       7.783392e+15
3                                       7.228952e+15
4                                       7.327444e+15

[5 rows x 119 columns]
```

[56]:
```
interaction_only_indices = [i for i, name in enumerate(poly.
 ↪get_feature_names_out(w_cols)) if ' ' in name]
w_tilde = w_tilde[:, interaction_only_indices]
interaction_feature_names = [name for name in poly.
 ↪get_feature_names_out(w_cols) if ' ' in name]
```

[57]:
```
x_dash3 = np.column_stack((d, d_p, w_tilde, fixed_effects))
```

[58]:
```
lasso_y9 = LassoCV(cv=10).fit(x_dash3,y)
```

[59]:
```
coefficients9 = lasso_y9.coef_
feature_names9 = ['d', 'd_p'] + interaction_feature_names + fixed_effect_cols
coef_df9 = pd.DataFrame({'Feature': feature_names9, 'Coefficient':␣
 ↪coefficients9})
```

[60]:
```
coef_df9.head()
```

[60]:
```
                     Feature  Coefficient
0                          d          0.0
1                        d_p          0.0
```

```
2    popestimate employment          0.0
3  popestimate aveFareTotal          0.0
4      popestimate VRHTotal          0.0
```

[61]:
```
non_zero_coefs9 = coef_df9[coef_df9['Coefficient'] != 0]
non_zero_coefs9
```

[61]:
```
                                              Feature   Coefficient
99  popestimate employment VRHTotal VOMSTotal VRMT…  3.496233e-32
```

12 Regression 10

[62]:
```
x_dash4 = np.column_stack((d, d_f, w_tilde, fixed_effects))
```

[63]:
```
lasso_y10 = LassoCV(cv=10).fit(x_dash4,y)
```

[64]:
```
coefficients10 = lasso_y10.coef_
feature_names10 = ['d', 'd_f'] + interaction_feature_names + fixed_effect_cols
coef_df10 = pd.DataFrame({'Feature': feature_names10, 'Coefficient':␣
 ↪coefficients10})
```

[65]:
```
coef_df10.head()
```

[65]:
```
                     Feature  Coefficient
0                          d          0.0
1                        d_f          0.0
2    popestimate employment          0.0
3  popestimate aveFareTotal          0.0
4      popestimate VRHTotal          0.0
```

[66]:
```
non_zero_coefs10 = coef_df10[coef_df10['Coefficient'] != 0]
non_zero_coefs10
```

[66]:
```
                                              Feature   Coefficient
99  popestimate employment VRHTotal VOMSTotal VRMT…  3.496233e-32
```

13 Regression 11

[67]:
```
#first Lasso of y on other covariates
lasso_y11 = LassoCV(cv=10).fit(x_dash3,y)
```

[68]:
```
selected_features_y11 = np.where(lasso_y11.coef_ != 0)[0]
```

[69]:
```
#second Lasso of d on other covariates
x_dash5 = np.column_stack((w_tilde, fixed_effects))
lasso_d11 = LassoCV(cv=10).fit(x_dash5,d)
```

```
selected_features_d11 = np.where(lasso_d11.coef_ != 0)[0]
```

[70]:
```
#second Lasso of d_p on other covariates
lasso_dp11 = LassoCV(cv=10).fit(x_dash5,d_p)
selected_features_dp11 = np.where(lasso_dp11.coef_ != 0)[0]
```

[71]:
```
#OLS of y on all other selected covariates
selected_features11 = np.union1d(np.union1d(selected_features_y11,␣
 ↪selected_features_d11), selected_features_dp11)
Z_selected11 = x_dash3[:, selected_features11]
X_final11 = np.column_stack((d,d_f, Z_selected11))
ols_model2 = sm.OLS(y, X_final11).fit()
ols_model2.summary()
```

```
/Users/hibafarhan/anaconda3/envs/quant_fin/lib/python3.12/site-
packages/statsmodels/regression/linear_model.py:1966: RuntimeWarning: divide by
zero encountered in scalar divide
  return np.sqrt(eigvals[0]/eigvals[-1])
```

[71]:

Dep. Variable:	UPTTotal	**R-squared:**	-36.573
Model:	OLS	**Adj. R-squared:**	-36.573
Method:	Least Squares	**F-statistic:**	-7.418e+04
Date:	Sat, 08 Jun 2024	**Prob (F-statistic):**	1.00
Time:	16:42:18	**Log-Likelihood:**	-2.9655e+05
No. Observations:	76213	**AIC:**	5.931e+05
Df Residuals:	76211	**BIC:**	5.931e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
x1	6.111e-37	1.47e-38	41.691	0.000	5.82e-37	6.4e-37
x2	-7.434e-31	1.78e-32	-41.691	0.000	-7.78e-31	-7.08e-31
x3	5e-18	1.2e-19	41.691	0.000	4.76e-18	5.24e-18
x4	-3.762e-32	5.48e-33	-6.870	0.000	-4.83e-32	-2.69e-32

Omnibus:	74379.108	**Durbin-Watson:**	0.001
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	10578284.462
Skew:	-4.407	**Prob(JB):**	0.00
Kurtosis:	60.039	**Cond. No.**	inf

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 0. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

[72]:
```
#After Double-LASSO variables and coefficients
treatment_vars = ['d', 'd_p']
interaction_var_names = poly.get_feature_names_out(w_cols)
fixed_effect_names = fixed_effect_cols
```

```
all_variable_names = treatment_vars + list(interaction_var_names) +␣
 ↪fixed_effect_names
coefficients11 = ols_model2.params
selected_variable_names11 = []
for idx in selected_features11:
    selected_variable_names11.append(all_variable_names[idx])
selected_variable_names11 = ['d', 'd_p'] + selected_variable_names11
result11 = pd.DataFrame({
    'Feature': selected_variable_names11,
    'Coefficient': coefficients11
})

result11
```

[72]:

```
                                     Feature   Coefficient
x1                                         d  6.111187e-37
x2                                       d_p -7.434010e-31
x3  aveFareTotal VRHTotal VRMTotal gasPrice  5.000041e-18
x4     VRHTotal VOMSTotal VRMTotal gasPrice -3.761656e-32
```

14 Regression 12

[73]:

```
#first Lasso of y on other covariates
lasso_y12 = LassoCV(cv=10).fit(x_dash4,y)
selected_features_y12 = np.where(lasso_y12.coef_ != 0)[0]
```

[74]:

```
#second Lasso of d on other covariates
lasso_d12 = LassoCV(cv=10).fit(x_dash5,d)
selected_features_d12 = np.where(lasso_d12.coef_ != 0)[0]
```

[75]:

```
#second Lasso of d_p on other covariates
lasso_df12 = LassoCV(cv=10).fit(x_dash5,d_f)
selected_features_df12 = np.where(lasso_df12.coef_ != 0)[0]
```

[76]:

```
#OLS of y on all other selected covariates
selected_features12 = np.union1d(np.union1d(selected_features_y12,␣
 ↪selected_features_d12), selected_features_df12)
Z_selected12 = x_dash4[:, selected_features12]
X_final12 = np.column_stack((d,d_f, Z_selected12))
ols_model3 = sm.OLS(y, X_final12).fit()
ols_model3.summary()
```

```
/Users/hibafarhan/anaconda3/envs/quant_fin/lib/python3.12/site-
packages/statsmodels/regression/linear_model.py:1966: RuntimeWarning: divide by
zero encountered in scalar divide
  return np.sqrt(eigvals[0]/eigvals[-1])
```

[76]:

Dep. Variable:	UPTTotal	**R-squared:**	-36.573
Model:	OLS	**Adj. R-squared:**	-36.573
Method:	Least Squares	**F-statistic:**	-7.418e+04
Date:	Sat, 08 Jun 2024	**Prob (F-statistic):**	1.00
Time:	16:44:11	**Log-Likelihood:**	-2.9655e+05
No. Observations:	76213	**AIC:**	5.931e+05
Df Residuals:	76211	**BIC:**	5.931e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
x1	6.111e-37	1.47e-38	41.691	0.000	5.82e-37	6.4e-37
x2	-7.434e-31	1.78e-32	-41.691	0.000	-7.78e-31	-7.08e-31
x3	5e-18	1.2e-19	41.691	0.000	4.76e-18	5.24e-18
x4	-3.762e-32	5.48e-33	-6.870	0.000	-4.83e-32	-2.69e-32

Omnibus:	74379.108	**Durbin-Watson:**	0.001
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	10578284.462
Skew:	-4.407	**Prob(JB):**	0.00
Kurtosis:	60.039	**Cond. No.**	inf

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 0. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

[77]:
```
dt.head()
```

[77]:
```
             UPTTotal  treatUberX  treatGTNotStd  popestimate  employment  \
dateSurvey
2004-01-01    8296756         0.0           0.00      3163703     1572859
2004-02-01    7847113         0.0           1.40      3163703     1581307
2004-03-01    9011399         0.0           3.00      3163703     1592152
2004-04-01    8656389         0.0           2.25      3163703     1598167
2004-05-01    8378406         0.0           2.60      3163703     1593356

            aveFareTotal  VRHTotal  VOMSTotal   VRMTotal  gasPrice  …  \
dateSurvey                                                          …
2004-01-01      0.778015  333329.0     2626.0  4740396.0     1.701  …
2004-02-01      0.778015  310535.0     2626.0  4398939.0     1.862  …
2004-03-01      0.778015  356761.0     2626.0  5176183.0     2.063  …
2004-04-01      0.778015  341191.0     2626.0  4889387.0     2.121  …
2004-05-01      0.778015  333418.0     2626.0  4747018.0     2.266  …

            agency_Yuma Metropolitan Planning Organization  \
dateSurvey
2004-01-01                                           False
2004-02-01                                           False
2004-03-01                                           False
```

```
2004-04-01                                           False
2004-05-01                                           False

            agency_vRide, Inc. - Anchorage  agency_vRide, Inc. - Atlanta  \
dateSurvey
2004-01-01                           False                         False
2004-02-01                           False                         False
2004-03-01                           False                         False
2004-04-01                           False                         False
2004-05-01                           False                         False

            agency_vRide, Inc. - Denver  agency_vRide, Inc. - Tucson  \
dateSurvey
2004-01-01                        False                        False
2004-02-01                        False                        False
2004-03-01                        False                        False
2004-04-01                        False                        False
2004-05-01                        False                        False

            agency_vRide, Inc. - Valley Metro  P_it  d_p  F_it  d_f
dateSurvey
2004-01-01                              False     1  0.0     1  0.0
2004-02-01                              False     1  0.0     1  0.0
2004-03-01                              False     1  0.0     1  0.0
2004-04-01                              False     1  0.0     1  0.0
2004-05-01                              False     1  0.0     1  0.0

[5 rows x 859 columns]
```

[78]:

```
#After Double-LASSO variables and coefficients
treatment_vars = ['d', 'd_f']
interaction_var_names = poly.get_feature_names_out(w_cols)
fixed_effect_names = fixed_effect_cols
all_variable_names = treatment_vars + list(interaction_var_names) +␣
 ↪fixed_effect_names
coefficients12 = ols_model3.params
selected_variable_names12 = []
for idx in selected_features12:
    selected_variable_names12.append(all_variable_names[idx])
selected_variable_names12 = ['d', 'd_f'] + selected_variable_names12
result12 = pd.DataFrame({
    'Feature': selected_variable_names12,
    'Coefficient': coefficients12
})

result12
```

[78]:
```
                                         Feature   Coefficient
x1                                             d  6.111187e-37
x2                                           d_f -7.434010e-31
x3  aveFareTotal VRHTotal VRMTotal gasPrice  5.000041e-18
x4     VRHTotal VOMSTotal VRMTotal gasPrice -3.761656e-32
```

The results for the Double-LASSO regressions when taking the interaction term as w tilde, we observe that the regression results are qualitatively different form the original paper. The paper observes that presence of UberX has a negative effect on number of commutes from public transport agency. However, our analysis shows that there exists a positve relationship between presence of UberX and number of commutes from public transport agency.

When taking the W as a vector of all other variables, we observe that the coefficient for presence of UberX penalizes for both cases of observing different dummy variables. The result in these cases are qualiitatively different than the original paper.