# Group

- Aliaksei Kanstantsinau
- Simon Geller
- Gabriel de La Noue

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.regression.linear_model import OLS
```

# Question 1: Fixed effects and within transformations

You will find a modified version the imports-85.csv (imports85_modified.csv) file attached to this assignment. Again, make sure that all continuous variables of interest are numeric.

In [2]:
```
data = pd.read_csv('data/imports85_modified-1.csv')
data['city.mpg']=pd.to_numeric(data['city.mpg'], errors='coerce')
data['horsepower'] = pd.to_numeric(data['horsepower'], errors='coerce')
data.head()
```

Out[2]:

| | symboling | normalized-losses | make | fuel.type | aspiration | num.of.doors | body.style | dr |
|---|---|---|---|---|---|---|---|---|
| **0** | 3 | NaN | alfa-romero | gas | std | two | convertible | |
| **1** | 3 | NaN | alfa-romero | gas | std | two | convertible | |
| **2** | 1 | NaN | alfa-romero | gas | std | two | hatchback | |
| **3** | 2 | 164.0 | audi | gas | std | four | sedan | |
| **4** | 2 | 164.0 | audi | gas | std | four | sedan | |

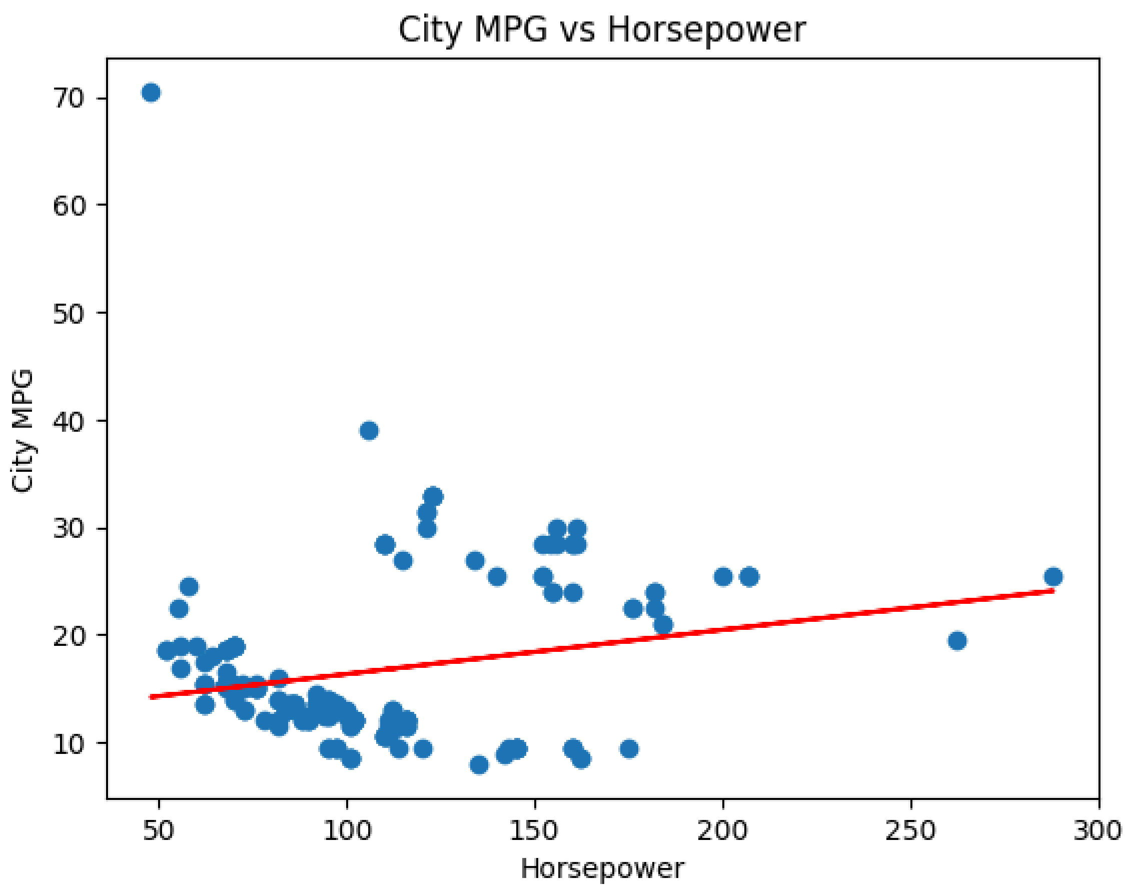
5 rows × 26 columns

1. Regress fuel efficiency (city.mpg) on horsepower without fixed effects. What would

you conclude based on that regression?

In [3]:
```
limited_df = data[['city.mpg', 'horsepower']].dropna()
y = limited_df['city.mpg']
X = limited_df['horsepower']
X = sm.add_constant(X)
model = OLS(y, X).fit()
# print(model.summary())

plt.scatter(limited_df['horsepower'], limited_df['city.mpg'])
plt.plot(limited_df['horsepower'], model.predict(X), color='red')
plt.xlabel('Horsepower')
plt.ylabel('City MPG')
plt.title('City MPG vs Horsepower')
plt.show()
```




Based on that visualization, it seems that there are two separate clusters, so a linear regression on this whole data without any pre treatment doesn't make sense statistically.

2. Repeat the same regression but this time, add a fixed effect for number of cylinders

being “two” or “four”. What would you conclude based on this new regression? What do you think drives the results in part 1?

In [4]:
```
limited_df['two_cylinders'] = (data['num.of.cylinders'] == 'two').astype(int
limited_df['four_cylinders'] = (data['num.of.cylinders'] == 'four').astype(i
```

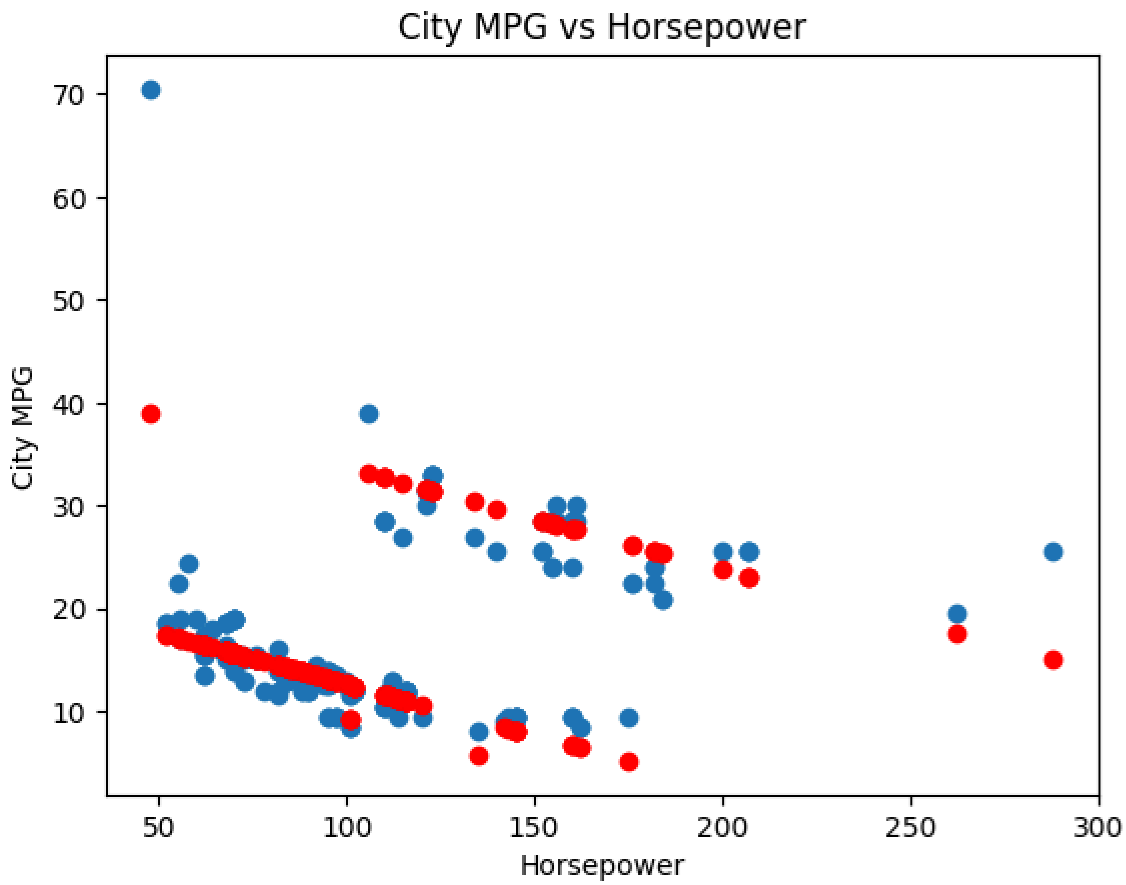
```
# limited_df['intersection'] = limited_df['two_cylinders'] * limited_df['fou
limited_df = limited_df.dropna()
# print(limited_df.head())
# print(limited_df['intersection'].sum())

X = limited_df[['horsepower', 'two_cylinders', 'four_cylinders']]
X = sm.add_constant(X)
y = limited_df['city.mpg']
model = OLS(y, X).fit()
print(model.summary())

plt.scatter(limited_df['horsepower'], limited_df['city.mpg'])
plt.scatter(limited_df['horsepower'], model.predict(X), color='red')
plt.xlabel('Horsepower')
plt.ylabel('City MPG')
plt.title('City MPG vs Horsepower')
plt.show()
```

```
                            OLS Regression Results
============================================================================
==
Dep. Variable:               city.mpg   R-squared:                       0.8
27
Model:                            OLS   Adj. R-squared:                  0.8
25
Method:                 Least Squares   F-statistic:                     31
7.7
Date:                Sun, 14 Apr 2024   Prob (F-statistic):           1.34e-
75
Time:                        18:01:42   Log-Likelihood:                -516.
05
No. Observations:                 203   AIC:                             104
0.
Df Residuals:                     199   BIC:                             105
3.
Df Model:                           3
Covariance Type:            nonrobust
============================================================================
======
                     coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------------
------
const             43.6865      1.229     35.559      0.000      41.264
46.109
horsepower        -0.0996      0.007    -13.645      0.000      -0.114
-0.085
two_cylinders    -24.4047      1.658    -14.715      0.000     -27.675     -
21.134
four_cylinders   -21.0808      0.716    -29.457      0.000     -22.492     -
19.670
============================================================================
==
Omnibus:                      269.395   Durbin-Watson:                   1.7
33
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             25082.7
81
Skew:                           5.515   Prob(JB):                         0.
00
Kurtosis:                      56.327   Cond. No.                         91
0.
============================================================================
==

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




This new regression fits the data way better. We can see it visually and also thanks to the p-values of the t-statistic.

Based on this regression, one could conclude that the relationship between city mpg and horsepower is different wether the engine has two or four cylinders.

3. (Within transformation) Now obtain the mean city.mpg and horsepower for each

group. Use these group means to demean horsepower and city.mpg. Run the same regression you ran in part 1. Are the results different? Are the results obtained here different from the results in part 2? What does this tell you about the relation between fixed effect regressions and within transformations?

In [5]:
```
two_cylinder_mean = limited_df.where(limited_df['two_cylinders'] == 1).dropr
four_cylinder_mean = limited_df.where(limited_df['four_cylinders'] == 1).drc
limited_df['city.mpg'] = limited_df['city.mpg'] - limited_df['two_cylinders'
hp_two_cylinder_mean = limited_df.where(limited_df['two_cylinders'] == 1).dr
hp_four_cylinder_mean = limited_df.where(limited_df['four_cylinders'] == 1).
limited_df['horsepower'] = limited_df['horsepower'] - limited_df['two_cylinc



X = limited_df[['horsepower', 'two_cylinders', 'four_cylinders']]
X = sm.add_constant(X)
y = limited_df['city.mpg']
```

```
model = OLS(y, X).fit()
model.summary()

# plt.scatter(limited_df['horsepower'], limited_df['city.mpg'])
# plt.scatter(limited_df['horsepower'], model.predict(X), color='red')
# plt.xlabel('Horsepower')
# plt.ylabel('City MPG')
# plt.title('City MPG vs Horsepower')
# plt.show()
```

Out[5]:

OLS Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | city.mpg | **R-squared:** | 0.937 |
| **Model:** | OLS | **Adj. R-squared:** | 0.936 |
| **Method:** | Least Squares | **F-statistic:** | 981.0 |
| **Date:** | Sun, 14 Apr 2024 | **Prob (F-statistic):** | 6.36e-119 |
| **Time:** | 18:01:43 | **Log-Likelihood:** | -516.05 |
| **No. Observations:** | 203 | **AIC:** | 1040. |
| **Df Residuals:** | 199 | **BIC:** | 1053. |
| **Df Model:** | 3 | | |
| **Covariance Type:** | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| **const** | 43.6865 | 1.229 | 35.559 | 0.000 | 41.264 | 46.109 |
| **horsepower** | -0.0996 | 0.007 | -13.645 | 0.000 | -0.114 | -0.085 |
| **two_cylinders** | -43.6865 | 1.980 | -22.065 | 0.000 | -47.591 | -39.782 |
| **four_cylinders** | -43.6865 | 1.253 | -34.857 | 0.000 | -46.158 | -41.215 |

| | | | |
|---|---|---|---|
| **Omnibus:** | 269.395 | **Durbin-Watson:** | 1.733 |
| **Prob(Omnibus):** | 0.000 | **Jarque-Bera (JB):** | 25082.781 |
| **Skew:** | 5.515 | **Prob(JB):** | 0.00 |
| **Kurtosis:** | 56.327 | **Cond. No.** | 836. |

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

We notice that the coefficients are equal and the R^2 is better (0.937 > 0.827). Fixed effect regressions and within transformations both achieve the same goal but they lead to different coefficients, which makes the interpretation of the regerssions different.

# Question 2: On marginal significance and trading strategy improvements

You come up with a signal of stock outperformance: log total asset growth. You realize that your professor has conveniently already coded up this variable for you in the dataset StockRetAcct_DT.csv. The variable is called "lnInv".

1. Using the Fama-MacBeth regression approach, what are the average return, standard

deviation and Sharpe ratio of the trading strategy implied by using only an intercept and lnInv on the right hand side in the regressions?

In [6]:
```
# Using the code from the class slides we can compute the statistics of this
StockRetAcct_DF = pd.read_csv('data/StockRetAcct_DT.csv')

# We compute the excess annual return to use for the regression after
StockRetAcct_DF['ExRet'] = np.exp(StockRetAcct_DF['lnAnnRet']) - np.exp(Stoc

# define function that returns OLS coefficients from fitted regression
def ols_coef(x,formula): return smf.ols(formula,data=x).fit().params
# the below runs regression ExRet ~ lnInv by year, including intercept
res = (StockRetAcct_DF.groupby('year').apply(ols_coef,'ExRet ~ lnInv'))
print('Mean Return: ' , str(res['lnInv'].mean())+'\n',
    'Std Dev:      ', str(res['lnInv'].std())+'\n',
    'Sharpe Ratio ' , str(res['lnInv'].mean()/
    res['lnInv'].std())+'\n',
    't-stat:      ' , str(35**.5*(res['lnInv'].mean())/
    res['lnInv'].std()), sep="\n")
```

```
Mean Return:
-0.08679146319781365

Std Dev:
0.14864410759416122

Sharpe Ratio
-0.5838876804641185

t-stat:
-3.4543261019947002
```

2. What is the analytical expression for the portfolio weights in this case? (I'm looking for

a formula)

$$w_{i, t-1} = \frac{1}{N} \frac{(x_{i, t-1}-\mathbb{E}_N[x_{i, t-1}])}{\mathbb{Var}_N[x_{i, t-1}]}$$
With :

- $x_{i, t-1}$ : Risk premium of the company $i$ at time $t$.
- $N$ : number of companies.

3. You worry that there is industry-related noise associated with the characteristic lnInv

and want to clean up your trading strategy with the goal of reducing exposure to unpriced industry risks. What regressions to you run? Report mean, standard deviation, and Sharpe ratio of the 'cleaned-up' trading strategy.

We run a similar regression but with industry dummies.

In [7]:
```
StockRetAcct_DF['Industry']=StockRetAcct_DF['ff_ind'].astype(object)
res = (StockRetAcct_DF.groupby('year').apply(ols_coef, 'ExRet ~ lnInv +Indus
print('Mean Returns: ', str(res.mean()[1:])+'\n',
    'Std Dev:      ', str(res.std()[1:])+'\n',
    'Sharpe Ratio ' , str(res.mean()[1:]/
    res.std()[1:])+'\n',
    't-stat:      ', str(35**.5*(res.mean()[1:])/
    res.std()[1:]), sep="\n")
```

```
Mean Returns:
Industry[T.2.0]    -0.012202
Industry[T.3.0]    -0.016387
Industry[T.4.0]    -0.029597
Industry[T.5.0]    -0.001101
Industry[T.6.0]     0.006305
Industry[T.7.0]     0.023252
Industry[T.8.0]    -0.018057
Industry[T.9.0]     0.008121
Industry[T.10.0]    0.029135
Industry[T.11.0]    0.012417
Industry[T.12.0]   -0.023147
lnInv              -0.082578
dtype: float64

Std Dev:
Industry[T.2.0]     0.130250
Industry[T.3.0]     0.129407
Industry[T.4.0]     0.297996
Industry[T.5.0]     0.098524
Industry[T.6.0]     0.280174
Industry[T.7.0]     0.217234
Industry[T.8.0]     0.143258
Industry[T.9.0]     0.093976
Industry[T.10.0]    0.184509
Industry[T.11.0]    0.118847
Industry[T.12.0]    0.090814
lnInv               0.101964
dtype: float64

Sharpe Ratio
Industry[T.2.0]    -0.093681
Industry[T.3.0]    -0.126628
Industry[T.4.0]    -0.099320
Industry[T.5.0]    -0.011177
Industry[T.6.0]     0.022504
Industry[T.7.0]     0.107039
Industry[T.8.0]    -0.126045
Industry[T.9.0]     0.086421
Industry[T.10.0]    0.157905
Industry[T.11.0]    0.104476
Industry[T.12.0]   -0.254884
lnInv              -0.809868
dtype: float64

t-stat:
Industry[T.2.0]    -0.554223
Industry[T.3.0]    -0.749142
Industry[T.4.0]    -0.587584
Industry[T.5.0]    -0.066126
Industry[T.6.0]     0.133133
Industry[T.7.0]     0.633249
Industry[T.8.0]    -0.745693
Industry[T.9.0]     0.511272
Industry[T.10.0]    0.934180
Industry[T.11.0]    0.618091
```

```
Industry[T.12.0]    -1.507913
lnInv               -4.791247
dtype: float64
```

4. As in the class notes, plot the cumulative returns to the simple and the 'cleaned-up'

trading strategies based on your new signal, lnInv. Make sure both trading strategies result in portfolios with a 15% return standard deviation.

## The function FamaMacBeth does not work for me, so I could not go further.

It always gives me the error : "ValueError: dependent and exog must have the same number of observations. The number of observations in dependent is 66788, and the number of observations in exog is 133576.".

Which is not true as `print(y.shape, X.shape)` gives `(66788, 1) (66788, 2)`. I have verified everything and the documentation does not give any information that could help. It should not be the function `dmatrices` as splitting the dataset by hand gives the same error message.

The following code normally displays the cumulative returns to the simple and the 'cleaned-up' trading strategies based on the new signal lnInv.

Both trading strategies result in portfolios with a 15% return standard deviation.

In [8]:
```
# from linearmodels.panel.model import FamaMacBeth
# from patsy import dmatrices
# y, X  = dmatrices('ExRet~lnInv', StockRetAcct_DF, return_type = 'dataframe
# res1 = FamaMacBeth(y, X).fit()

# LastDate = StockRetAcct_DF[StockRetAcct_DF['year']==2014].dropna()
# LastDate = LastDate[['lnInv']].assign(c=1).sort_index(axis=1)
# # Create dummy variables for each industry. We drop the last column becaus
# # include a constant to avoid multicollinearity
# StockRetAcct_DF['Industry']=StockRetAcct_DF['ff_ind'].astype(object)
# LastDate = LastDate.join(pd.get_dummies(StockRetAcct_DF.Industry).iloc[:,:
# # use (X'X)^(-1) X' formula to compute weights
# portweights_lnInv = np.matmul(np.linalg.inv(np.matmul(LastDate.transpose()
# values,LastDate.values)),LastDate.transpose().values)
# # lnInv is third row
# portweights_lnInv = np.matmul(np.array([1,0,0,0,0,0,0,0,0,0,0,0,0]).T, por
# lnInvstdev = res.lnInv.std()
# lnInvret = res.lnInv
# # scale portfolio weights to get 15% standard deviation of returns
# portweights_lnInv = portweights_lnInv*0.15/lnInvstdev

# y, X = dmatrices('ExRet~lnInv+Industry', StockRetAcct_DF, return_type = 'c
# res  = FamaMacBeth(y,X).fit()

# # for plotting, get the scaled excess portfolio returns
```

```
# lnInv_ret = lnInvret*0.15/lnInvstdev
# # create cumulative log return series
# cum_ret_lnInv = pd.DataFrame.cumsum(np.log(1+lnInv_ret))
# # get "old" simple value strategy returns
# lnInvstdev = res1.lnInv.std()
# lnInvret = res1.lnInv
# lnInv_old_ret = pd.DataFrame.cumsum(np.log(1+lnInvret*0.15/lnInvstdev))
# summary = pd.DataFrame()
# summary['NewValue'] = np.exp(cum_ret_lnInv)
# summary['OldValue'] = np.exp(lnInv_old_ret)
# # Plot Old Value vs New Value
# sns.set_style('darkgrid')
# plt.figure()
# ax=sns.lineplot(data=summary,dashes = False,linewidth = 3)
# ax.set(xlabel = 'Year',
#     ylabel = 'Cumulative Return',
#     title  = "Old Value vs New Value")
```

# Question 3: Predicting medium to long-run firm-level return variance

There are many return volatility models, such as GARCH. These work best at shorter horizons. As an alternative, we will explore a panel regression approach to predicting firm- level return variance. The data set StockRetAcct_DT.csv has annual realized variance (rv), calculated as the sum of squared daily returns to each firm, each year. Run panel forecasting regressions to forecast firm-level one-year ahead rv along the lines of what we did with lnROE in class.

1. Try with and without industry and year fixed effects, with and without clustering of

standard errors. Discuss which specification makes most sense to you. In particular, discuss the effect of a year fixed effect. What is the intuition for the impact of this fixed effect?

In [9]:
```
retData = pd.read_csv('data/StockRetAcct_DT.csv')
#Vanilla, fixed-effect free model
X = retData['rv'].shift(1)[1:].fillna(0)
X = sm.add_constant(X)
y = retData['rv'][1:]
vanilla_model = OLS(y, X).fit()
print(vanilla_model.summary())
```

```
                            OLS Regression Results
==============================================================================
==
Dep. Variable:                     rv   R-squared:                       n
an
Model:                            OLS   Adj. R-squared:                  n
an
Method:                 Least Squares   F-statistic:                     n
an
Date:                Sun, 14 Apr 2024   Prob (F-statistic):              n
an
Time:                        18:01:43   Log-Likelihood:                  n
an
No. Observations:               70755   AIC:                             n
an
Df Residuals:                   70753   BIC:                             n
an
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
==
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const             nan        nan        nan        nan        nan          n
an
rv                nan        nan        nan        nan        nan          n
an
==============================================================================
==
Omnibus:                          nan   Durbin-Watson:                   n
an
Prob(Omnibus):                    nan   Jarque-Bera (JB):                n
an
Skew:                             nan   Prob(JB):                        n
an
Kurtosis:                         nan   Cond. No.                       5.
48
==============================================================================
==

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

In [10]:

```
inds = retData['ff_ind'].dropna().unique()
regression_df = retData[['year', 'rv']]
regression_df['lag_rv'] = regression_df['rv'].shift(1).fillna(0)

for year in regression_df['year'].unique():
    regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(

regression_df['ff_ind'] = retData['ff_ind'].fillna(0)
for ind in inds:
    regression_df['is_ind_'+str(ind)] = (regression_df['ff_ind'] == ind).ast
```

```
regression_df.head()
# print(regression_df.head())

year_fe_features = regression_df[['lag_rv'] + [('is_' + str(year)) for year
year_fe_features = sm.add_constant(year_fe_features)
y = regression_df['rv'].fillna(0)
year_fe_model = OLS(y, year_fe_features).fit()
print(year_fe_model.summary())
```

```
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:3: 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
  regression_df['lag_rv'] = regression_df['rv'].shift(1).fillna(0)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
/var/folders/r0/zk7h1dpx693gc15c5y61js8w0000gn/T/ipykernel_89533/2185439882.
py:6: 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
  regression_df['is_'+str(year)] = (regression_df['year'] == year).astype(in
t)
```

```
                            OLS Regression Results
==============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.4
67
Model:                            OLS   Adj. R-squared:                  0.4
67
Method:                 Least Squares   F-statistic:                     172
4.
Date:                Sun, 14 Apr 2024   Prob (F-statistic):               0.
00
Time:                        18:01:44   Log-Likelihood:                 4045
7.
No. Observations:               70756   AIC:                       -8.084e+
04
Df Residuals:                   70719   BIC:                       -8.050e+
04
Df Model:                          36
Covariance Type:            nonrobust
==============================================================================
==
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const      -1.026e+10    6.4e+09     -1.602      0.109   -2.28e+10    2.29e+
09
lag_rv         0.3522      0.003    110.589      0.000       0.346       0.3
58
is_1980     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1981     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1982     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1983     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1991     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_2000     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1986     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1987     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1988     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1989     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1990     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1992     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1993     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
10
is_1994     1.026e+10    6.4e+09      1.602      0.109   -2.29e+09    2.28e+
```

```
10
is_1996      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_1997      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_1998      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_1999      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2001      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2002      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2008      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2009      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2010      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2011      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2012      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2013      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2014      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2003      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2004      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2005      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2006      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_2007      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_1995      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_1984      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
is_1985      1.026e+10    6.4e+09      1.602       0.109   -2.29e+09    2.28e+
10
=============================================================================
==
Omnibus:                    32775.551   Durbin-Watson:                   1.9
86
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           565891.8
72
Skew:                           1.800   Prob(JB):                         0.
00
Kurtosis:                      16.379   Cond. No.                      7.69e+
13
=============================================================================
==
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
[2] The smallest eigenvalue is 1.26e-23. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

In [11]:
```
ind_fe_features = regression_df[['lag_rv'] + [('is_ind_' + str(ind)) for inc
ind_fe_features = sm.add_constant(ind_fe_features)
y = regression_df['rv'].fillna(0)
ind_fe_model = OLS(y, ind_fe_features).fit()
print(ind_fe_model.summary())
```

```
                            OLS Regression Results
==============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.1
89
Model:                            OLS   Adj. R-squared:                  0.1
89
Method:                 Least Squares   F-statistic:                     127
2.
Date:                Sun, 14 Apr 2024   Prob (F-statistic):               0.
00
Time:                        18:01:44   Log-Likelihood:                 2560
1.
No. Observations:               70756   AIC:                        -5.117e+
04
Df Residuals:                   70742   BIC:                        -5.105e+
04
Df Model:                          13
Covariance Type:            nonrobust
==============================================================================
===
                 coef    std err          t      P>|t|      [0.025      0.9
75]
------------------------------------------------------------------------------
---
const         -0.0413      0.069     -0.601      0.548      -0.176       0.
094
lag_rv         0.3438      0.003     98.286      0.000       0.337       0.
351
is_ind_3.0     0.1318      0.069      1.915      0.055      -0.003       0.
267
is_ind_10.0    0.1821      0.069      2.646      0.008       0.047       0.
317
is_ind_6.0     0.2141      0.069      3.110      0.002       0.079       0.
349
is_ind_12.0    0.1474      0.069      2.142      0.032       0.013       0.
282
is_ind_11.0    0.1161      0.069      1.687      0.092      -0.019       0.
251
is_ind_1.0     0.1129      0.069      1.639      0.101      -0.022       0.
248
is_ind_9.0     0.1434      0.069      2.084      0.037       0.009       0.
278
is_ind_7.0     0.1595      0.069      2.315      0.021       0.024       0.
295
is_ind_2.0     0.1324      0.069      1.922      0.055      -0.003       0.
267
is_ind_5.0     0.1181      0.069      1.714      0.086      -0.017       0.
253
is_ind_8.0     0.0810      0.069      1.176      0.240      -0.054       0.
216
is_ind_4.0     0.1569      0.069      2.279      0.023       0.022       0.
292
==============================================================================
==
Omnibus:                    49472.965   Durbin-Watson:                   2.0
```

```
48
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1008799.9
98
Skew:                           3.142   Prob(JB):                         0.
00
Kurtosis:                      20.398   Cond. No.                         41
8.
==============================================================================
==

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

In [12]:

```
joint_fe_features = regression_df[['lag_rv'] + [('is_' + str(year)) for year
joint_fe_features.head()
joint_fe_features = sm.add_constant(joint_fe_features)
y = regression_df['rv'].fillna(0)
joint_fe_model = OLS(y, joint_fe_features).fit()
print(joint_fe_model.summary())
```

```
                            OLS Regression Results
============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.4
67
Model:                            OLS   Adj. R-squared:                  0.4
67
Method:                 Least Squares   F-statistic:                     154
9.
Date:                Sun, 14 Apr 2024   Prob (F-statistic):               0.
00
Time:                        18:01:45   Log-Likelihood:                 4043
8.
No. Observations:               70756   AIC:                       -8.079e+
04
Df Residuals:                   70715   BIC:                       -8.042e+
04
Df Model:                          40
Covariance Type:            nonrobust
============================================================================
==
                 coef    std err          t      P>|t|      [0.025      0.97
5]
----------------------------------------------------------------------------
--
const      -5.634e+10   8.78e+10     -0.641      0.521   -2.28e+11    1.16e+
11
lag_rv         0.3542      0.004     98.888      0.000       0.347       0.3
61
is_1980      5.97e+09   1.51e+10      0.394      0.693   -2.37e+10    3.56e+
10
is_1981     1.353e+09   1.66e+10      0.081      0.935   -3.12e+10    3.39e+
10
is_1982     5.258e+10   3.06e+10      1.719      0.086   -7.36e+09    1.13e+
11
is_1983     3.705e+11   1.01e+11      3.658      0.000    1.72e+11    5.69e+
11
is_1991     7.668e+10   2.71e+10      2.826      0.005    2.35e+10     1.3e+
11
is_2000     2.488e+10   4.77e+10      0.522      0.602   -6.86e+10    1.18e+
11
is_1986    -1.341e+10   7.33e+10     -0.183      0.855   -1.57e+11     1.3e+
11
is_1987    -4.516e+09   4.89e+10     -0.092      0.926      -1e+11    9.13e+
10
is_1988    -6.716e+09   6.28e+10     -0.107      0.915    -1.3e+11    1.16e+
11
is_1989    -4.022e+10   3.48e+10     -1.155      0.248   -1.08e+11     2.8e+
10
is_1990     2.187e+10   6.62e+10      0.330      0.741   -1.08e+11    1.52e+
11
is_1992     8.361e+08   2.33e+10      0.036      0.971   -4.48e+10    4.65e+
10
is_1993    -2.242e+10   5.44e+10     -0.412      0.680   -1.29e+11    8.42e+
10
is_1994     1.522e+10    3.6e+10      0.423      0.673   -5.53e+10    8.58e+
```

```
10
is_1996    8.207e+09   2.14e+10      0.384      0.701   -3.37e+10    5.01e+
10
is_1997   -1.039e+10   2.19e+10     -0.474      0.636   -5.33e+10    3.26e+
10
is_1998    4.011e+10   3.13e+10      1.282      0.200   -2.12e+10    1.01e+
11
is_1999   -4.821e+10   7.18e+10     -0.672      0.502   -1.89e+11    9.25e+
10
is_2001    6.471e+10   5.38e+10      1.202      0.229   -4.08e+10     1.7e+
11
is_2002    7.851e+10   7.75e+10      1.014      0.311   -7.33e+10     2.3e+
11
is_2008    1.419e+08   3.47e+11      0.000      1.000   -6.81e+11    6.81e+
11
is_2009    2.784e+10   4.45e+10      0.626      0.531   -5.93e+10    1.15e+
11
is_2010    2.758e+10   9.11e+10      0.303      0.762   -1.51e+11    2.06e+
11
is_2011   -2.569e+10   1.05e+11     -0.244      0.807   -2.32e+11    1.81e+
11
is_2012    2.903e+10   4.37e+10      0.664      0.506   -5.66e+10    1.15e+
11
is_2013   -3.896e+10    9.6e+10     -0.406      0.685   -2.27e+11    1.49e+
11
is_2014     -9.1e+10   9.52e+10     -0.956      0.339   -2.78e+11    9.56e+
10
is_2003     2.88e+10   4.36e+10      0.661      0.509   -5.67e+10    1.14e+
11
is_2004   -7.392e+10   6.34e+10     -1.166      0.244   -1.98e+11    5.04e+
10
is_2005    3.461e+11   2.39e+11      1.450      0.147   -1.22e+11    8.14e+
11
is_2006    3.038e+10   4.34e+10      0.701      0.484   -5.46e+10    1.15e+
11
is_2007   -2.544e+11   2.28e+11     -1.115      0.265   -7.02e+11    1.93e+
11
is_1995   -5.173e+10   1.06e+11     -0.488      0.625   -2.59e+11    1.56e+
11
is_1984     3.34e+10   4.45e+10      0.751      0.453   -5.38e+10    1.21e+
11
is_1985    2.803e+10    4.4e+10      0.637      0.524   -5.82e+10    1.14e+
11
is_1980    5.037e+10   8.04e+10      0.626      0.531   -1.07e+11    2.08e+
11
is_1981    5.498e+10   8.92e+10      0.617      0.538    -1.2e+11     2.3e+
11
is_1982    3.757e+09    7.9e+10      0.048      0.962   -1.51e+11    1.59e+
11
is_1983   -3.142e+11   1.65e+11     -1.902      0.057   -6.38e+11    9.58e+
09
is_1991   -2.035e+10   7.44e+10     -0.274      0.784   -1.66e+11    1.25e+
11
is_2000    3.146e+10   4.21e+10      0.748      0.455    -5.1e+10    1.14e+
11
is_1986    6.975e+10   1.05e+11      0.663      0.508   -1.37e+11    2.76e+
```

```
11
is_1987     6.085e+10   1.09e+11      0.561      0.575   -1.52e+11    2.74e+
11
is_1988     6.305e+10   1.35e+11      0.467      0.641   -2.02e+11    3.28e+
11
is_1989     9.656e+10   7.75e+10      1.246      0.213   -5.53e+10    2.48e+
11
is_1990     3.446e+10   8.11e+10      0.425      0.671   -1.25e+11    1.93e+
11
is_1992      5.55e+10   9.98e+10      0.556      0.578    -1.4e+11    2.51e+
11
is_1993     7.876e+10   8.55e+10      0.921      0.357   -8.89e+10    2.46e+
11
is_1994     4.112e+10   8.05e+10      0.511      0.609   -1.17e+11    1.99e+
11
is_1996     4.813e+10   7.96e+10      0.605      0.545   -1.08e+11    2.04e+
11
is_1997     6.672e+10   9.34e+10      0.714      0.475   -1.16e+11     2.5e+
11
is_1998     1.623e+10   8.67e+10      0.187      0.852   -1.54e+11    1.86e+
11
is_1999     1.045e+11   9.53e+10      1.097      0.272   -8.22e+10    2.91e+
11
is_2001    -8.372e+09   8.84e+10     -0.095      0.925   -1.82e+11    1.65e+
11
is_2002    -2.217e+10    1.1e+11     -0.202      0.840   -2.37e+11    1.93e+
11
is_2008     5.619e+10   3.48e+11      0.161      0.872   -6.26e+11    7.39e+
11
is_2009      2.85e+10   4.34e+10      0.656      0.512   -5.66e+10    1.14e+
11
is_2010     2.876e+10   1.14e+11      0.253      0.800   -1.94e+11    2.51e+
11
is_2011     8.202e+10   1.31e+11      0.628      0.530   -1.74e+11    3.38e+
11
is_2012      2.73e+10   4.42e+10      0.618      0.536   -5.93e+10    1.14e+
11
is_2013     9.529e+10   1.28e+11      0.745      0.456   -1.55e+11    3.46e+
11
is_2014     1.473e+11   1.32e+11      1.120      0.263    -1.1e+11    4.05e+
11
is_2003     2.753e+10   4.43e+10      0.622      0.534   -5.93e+10    1.14e+
11
is_2004     1.303e+11   1.05e+11      1.242      0.214   -7.53e+10    3.36e+
11
is_2005    -2.898e+11   2.42e+11     -1.196      0.232   -7.65e+11    1.85e+
11
is_2006     2.596e+10   4.51e+10      0.576      0.565   -6.24e+10    1.14e+
11
is_2007     3.107e+11   2.31e+11      1.344      0.179   -1.42e+11    7.64e+
11
is_1995     1.081e+11   1.29e+11      0.837      0.402   -1.45e+11    3.61e+
11
is_1984     2.294e+10   4.42e+10      0.519      0.604   -6.37e+10     1.1e+
11
is_1985      2.83e+10   4.39e+10      0.645      0.519   -5.77e+10    1.14e+
```

```
11
==============================================================================
==
Omnibus:                    32923.076   Durbin-Watson:                   1.9
91
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           570589.7
55
Skew:                           1.810   Prob(JB):                          0.
00
Kurtosis:                      16.433   Cond. No.                      1.71e+
16
==============================================================================
==

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
[2] The smallest eigenvalue is 2.63e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

Across regressions, we see pretty consistent t+1 autocorrelation of returns represented by the lag_rv coef whcih hovers around 35% (just under). Additionally, the is_year variables help in extracting out the realized mean bump to vol in that year with the is_ind_x variable doing the same for industry effects. Which we want to keep comes down to how parsimonious we would like our model to be and is up to the user

2. Also try forecasting at the 5-year horizon (rv in 5 years). How do the results change?

Can we predict return variance 5-years ahead? Is the 5-year lagged rv significant, or are other variables more important?

In [13]:
```
regression_df['5y_lag_rv'] = regression_df['rv'].shift(5).fillna(0)
vanilla_features_5y = sm.add_constant(regression_df['5y_lag_rv'])
year_fe_features_5y = sm.add_constant(regression_df[['5y_lag_rv'] + [('is_'
ind_fe_features_5y = sm.add_constant(regression_df[['5y_lag_rv'] + [('is_ind
joint_fe_features_5y = sm.add_constant(regression_df[['5y_lag_rv'] + [('is_'
y = regression_df['rv'].fillna(0)
vanilla_model_5y = OLS(y, vanilla_features_5y).fit()
year_fe_model_5y = OLS(y, year_fe_features_5y).fit()
ind_fe_model_5y = OLS(y, ind_fe_features_5y).fit()
joint_fe_model_5y = OLS(y, joint_fe_features_5y).fit()

print(vanilla_model_5y.summary())
```

```
                            OLS Regression Results
==============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.0
13
Model:                            OLS   Adj. R-squared:                  0.0
13
Method:                 Least Squares   F-statistic:                     90
1.7
Date:                Sun, 14 Apr 2024   Prob (F-statistic):          7.30e-1
97
Time:                        18:01:45   Log-Likelihood:                 1862
0.
No. Observations:               70756   AIC:                       -3.724e+
04
Df Residuals:                   70754   BIC:                       -3.722e+
04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
==
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const          0.1407      0.001    153.553      0.000       0.139       0.1
42
5y_lag_rv      0.1122      0.004     30.028      0.000       0.105       0.1
19
==============================================================================
==
Omnibus:                    48804.154   Durbin-Watson:                   1.2
60
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           740413.3
23
Skew:                           3.205   Prob(JB):                         0.
00
Kurtosis:                      17.493   Cond. No.                         5.
48
==============================================================================
==

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

In [14]:
```
print(year_fe_model_5y.summary())
```

```
                            OLS Regression Results
==============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.3
91
Model:                            OLS   Adj. R-squared:                  0.3
91
Method:                 Least Squares   F-statistic:                     126
3.
Date:                Sun, 14 Apr 2024   Prob (F-statistic):               0.
00
Time:                        18:01:45   Log-Likelihood:                 3573
9.
No. Observations:               70756   AIC:                         -7.140e+
04
Df Residuals:                   70719   BIC:                         -7.107e+
04
Df Model:                          36
Covariance Type:            nonrobust
==============================================================================
==
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const       1.452e+10   6.15e+09      2.362      0.018    2.47e+09    2.66e+
10
5y_lag_rv      0.1349      0.003     43.275      0.000       0.129       0.1
41
is_1980    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1981    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1982    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1983    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1991    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2000    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1986    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1987    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1988    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1989    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1990    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1992    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1993    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1994    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
```

```
09
is_1996    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1997    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1998    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1999    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2001    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2002    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2008    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2009    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2010    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2011    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2012    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2013    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2014    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2003    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2004    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2005    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2006    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_2007    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1995    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1984    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
is_1985    -1.452e+10   6.15e+09     -2.362      0.018   -2.66e+10   -2.47e+
09
============================================================================
==
Omnibus:                      30146.156   Durbin-Watson:                   1.2
18
Prob(Omnibus):                    0.000   Jarque-Bera (JB):          336228.3
35
Skew:                             1.747   Prob(JB):                        0.
00
Kurtosis:                        13.091   Cond. No.                     6.90e+
13
============================================================================
==
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
[2] The smallest eigenvalue is 1.57e-23. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

In [15]:
```
print(ind_fe_model_5y.summary())
```

```
                            OLS Regression Results                            
==============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.0
82
Model:                            OLS   Adj. R-squared:                  0.0
82
Method:                 Least Squares   F-statistic:                     48
5.2
Date:                Sun, 14 Apr 2024   Prob (F-statistic):               0.
00
Time:                        18:01:45   Log-Likelihood:                2119
4.
No. Observations:               70756   AIC:                        -4.236e+
04
Df Residuals:                   70742   BIC:                        -4.223e+
04
Df Model:                          13                                         
Covariance Type:            nonrobust                                         
==============================================================================
===
                  coef    std err          t      P>|t|      [0.025      0.9
75]
------------------------------------------------------------------------------
---
const          -0.0049      0.073     -0.067      0.947      -0.148       0.
139
5y_lag_rv       0.0573      0.004     15.569      0.000       0.050       0.
065
is_ind_3.0      0.1357      0.073      1.853      0.064      -0.008       0.
279
is_ind_10.0     0.2055      0.073      2.805      0.005       0.062       0.
349
is_ind_6.0      0.2483      0.073      3.390      0.001       0.105       0.
392
is_ind_12.0     0.1583      0.073      2.161      0.031       0.015       0.
302
is_ind_11.0     0.1139      0.073      1.556      0.120      -0.030       0.
257
is_ind_1.0      0.1090      0.073      1.487      0.137      -0.035       0.
253
is_ind_9.0      0.1534      0.073      2.094      0.036       0.010       0.
297
is_ind_7.0      0.1715      0.073      2.340      0.019       0.028       0.
315
is_ind_2.0      0.1368      0.073      1.865      0.062      -0.007       0.
281
is_ind_5.0      0.1157      0.073      1.579      0.114      -0.028       0.
259
is_ind_8.0      0.0626      0.073      0.855      0.393      -0.081       0.
206
is_ind_4.0      0.1719      0.073      2.345      0.019       0.028       0.
316
==============================================================================
==
Omnibus:                    48570.035   Durbin-Watson:                   1.3
```

```
25
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          793065.1
78
Skew:                           3.150   Prob(JB):                        0.
00
Kurtosis:                      18.143   Cond. No.                        41
8.
=============================================================================
==

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

In [16]:

```
print(joint_fe_model_5y.summary())
```

```
                            OLS Regression Results
============================================================================
==
Dep. Variable:                     rv   R-squared:                       0.3
91
Model:                            OLS   Adj. R-squared:                  0.3
91
Method:                 Least Squares   F-statistic:                     116
4.
Date:                Sun, 14 Apr 2024   Prob (F-statistic):               0.
00
Time:                        18:01:45   Log-Likelihood:                 3571
9.
No. Observations:               70756   AIC:                      -7.136e+
04
Df Residuals:                   70716   BIC:                      -7.099e+
04
Df Model:                          39
Covariance Type:            nonrobust
============================================================================
==
                 coef    std err          t      P>|t|      [0.025      0.97
5]
----------------------------------------------------------------------------
--
const       1.734e+10   5.17e+10      0.335      0.737   -8.41e+10    1.19e+
11
5y_lag_rv      0.1347      0.003     40.717      0.000       0.128       0.1
41
is_1980     5.213e+10   9.85e+10      0.530      0.596   -1.41e+11    2.45e+
11
is_1981    -1.265e+09   1.07e+10     -0.119      0.905   -2.21e+10    1.96e+
10
is_1982    -2.149e+10   1.42e+10     -1.516      0.129   -4.93e+10    6.29e+
09
is_1983    -8.792e+10   1.89e+10     -4.651      0.000   -1.25e+11   -5.09e+
10
is_1991    -2.497e+10    3.8e+10     -0.658      0.511   -9.94e+10    4.94e+
10
is_2000     1.321e+10   7.72e+10      0.171      0.864   -1.38e+11    1.64e+
11
is_1986    -2.513e+10   6.75e+10     -0.372      0.710   -1.57e+11    1.07e+
11
is_1987     9.591e+09   3.78e+10      0.254      0.799   -6.44e+10    8.36e+
10
is_1988    -4.483e+10   7.38e+10     -0.607      0.544    -1.9e+11    9.99e+
10
is_1989    -3.389e+10   2.02e+10     -1.681      0.093   -7.34e+10    5.62e+
09
is_1990     2.743e+10   5.77e+10      0.475      0.635   -8.58e+10    1.41e+
11
is_1992    -1.512e+10   2.84e+10     -0.532      0.595   -7.09e+10    4.06e+
10
is_1993     2.307e+10   3.01e+10      0.766      0.444    -3.6e+10    8.21e+
10
is_1994     1.227e+10   7.17e+10      0.171      0.864   -1.28e+11    1.53e+
```

```
11
is_1996     1.056e+10   4.59e+10      0.230      0.818   -7.93e+10      1e+
11
is_1997     1.251e+10   2.96e+10      0.423      0.672   -4.54e+10   7.05e+
10
is_1998     3.005e+10   2.54e+10      1.182      0.237   -1.98e+10   7.99e+
10
is_1999     5.959e+10   5.03e+10      1.185      0.236    -3.9e+10   1.58e+
11
is_2001     1.368e+11   1.42e+11      0.964      0.335   -1.41e+11   4.15e+
11
is_2002    -7.705e+10   5.52e+10     -1.395      0.163   -1.85e+11   3.12e+
10
is_2008     1.915e+10   7.46e+10      0.257      0.797   -1.27e+11   1.65e+
11
is_2009    -6.881e+10   7.84e+10     -0.878      0.380   -2.22e+11   8.49e+
10
is_2010     2.621e+10    1.6e+11      0.164      0.870   -2.87e+11    3.4e+
11
is_2011      -9.4e+09   2.58e+10     -0.364      0.716      -6e+10   4.12e+
10
is_2012     7.275e+09   1.39e+11      0.052      0.958   -2.66e+11    2.8e+
11
is_2013     1.096e+10   1.18e+11      0.093      0.926   -2.21e+11   2.43e+
11
is_2014    -1.452e+11   2.32e+11     -0.626      0.531      -6e+11   3.09e+
11
is_2003    -1.845e+11    2.1e+11     -0.879      0.380   -5.96e+11   2.27e+
11
is_2004    -6.672e+10    7.4e+10     -0.902      0.367   -2.12e+11   7.83e+
10
is_2005     3.413e+11   1.82e+11      1.876      0.061   -1.53e+10   6.98e+
11
is_2006    -9.533e+09   1.44e+11     -0.066      0.947   -2.92e+11   2.73e+
11
is_2007    -1.546e+11   1.59e+11     -0.975      0.330   -4.65e+11   1.56e+
11
is_1995    -1.374e+11   1.94e+11     -0.710      0.478   -5.17e+11   2.42e+
11
is_1984    -1.611e+10   2.77e+10     -0.581      0.561   -7.05e+10   3.83e+
10
is_1985    -8.413e+09   2.59e+10     -0.325      0.745   -5.91e+10   4.23e+
10
is_1980    -6.948e+10   9.63e+10     -0.721      0.471   -2.58e+11   1.19e+
11
is_1981    -1.608e+10   4.98e+10     -0.323      0.747   -1.14e+11   8.14e+
10
is_1982     4.144e+09   4.65e+10      0.089      0.929    -8.7e+10   9.53e+
10
is_1983     7.058e+10   4.96e+10      1.423      0.155   -2.67e+10   1.68e+
11
is_1991     7.627e+09   5.23e+10      0.146      0.884   -9.49e+10    1.1e+
11
is_2000    -3.055e+10   8.03e+10     -0.380      0.704   -1.88e+11   1.27e+
11
is_1986     7.786e+09   7.48e+10      0.104      0.917   -1.39e+11   1.54e+
```

```
11
is_1987    -2.693e+10   3.58e+10     -0.753      0.451    -9.7e+10    4.32e+
10
is_1988     2.749e+10   9.89e+10      0.278      0.781   -1.66e+11    2.21e+
11
is_1989     1.655e+10   5.47e+10      0.302      0.762   -9.07e+10    1.24e+
11
is_1990    -4.477e+10   6.78e+10     -0.660      0.509   -1.78e+11    8.82e+
10
is_1992    -2.221e+09   5.45e+10     -0.041      0.967   -1.09e+11    1.05e+
11
is_1993    -4.041e+10    5.4e+10     -0.748      0.455   -1.46e+11    6.55e+
10
is_1994    -2.961e+10   9.06e+10     -0.327      0.744   -2.07e+11    1.48e+
11
is_1996     -2.79e+10   6.88e+10     -0.406      0.685   -1.63e+11    1.07e+
11
is_1997    -2.985e+10   5.89e+10     -0.506      0.613   -1.45e+11    8.57e+
10
is_1998     -4.74e+10   5.36e+10     -0.884      0.377   -1.52e+11    5.76e+
10
is_1999    -7.693e+10   6.41e+10     -1.201      0.230   -2.02e+11    4.86e+
10
is_2001    -1.542e+11   1.46e+11     -1.056      0.291    -4.4e+11    1.32e+
11
is_2002     5.971e+10   6.89e+10      0.867      0.386   -7.54e+10    1.95e+
11
is_2008     -3.65e+10   8.57e+10     -0.426      0.670   -2.04e+11    1.31e+
11
is_2009     5.146e+10   8.99e+10      0.572      0.567   -1.25e+11    2.28e+
11
is_2010    -4.355e+10   1.61e+11     -0.270      0.787    -3.6e+11    2.73e+
11
is_2011    -7.942e+09    2.6e+10     -0.305      0.760   -5.89e+10     4.3e+
10
is_2012    -2.462e+10   1.42e+11     -0.174      0.862   -3.02e+11    2.53e+
11
is_2013     -2.83e+10   1.23e+11     -0.230      0.818   -2.69e+11    2.13e+
11
is_2014     1.279e+11   2.33e+11      0.550      0.582   -3.28e+11    5.84e+
11
is_2003     1.672e+11   2.11e+11      0.793      0.428   -2.46e+11     5.8e+
11
is_2004     4.937e+10   8.56e+10      0.577      0.564   -1.18e+11    2.17e+
11
is_2005    -3.586e+11   1.83e+11     -1.965      0.049   -7.16e+11   -8.81e+
08
is_2006     -7.81e+09   1.46e+11     -0.053      0.957   -2.94e+11    2.79e+
11
is_2007     1.372e+11    1.6e+11      0.858      0.391   -1.76e+11    4.51e+
11
is_1995       1.2e+11   1.94e+11      0.618      0.536   -2.61e+11    5.01e+
11
is_1984    -1.235e+09   2.87e+10     -0.043      0.966   -5.75e+10     5.5e+
10
is_1985    -8.929e+09   2.59e+10     -0.345      0.730   -5.97e+10    4.18e+
```

```
10
==============================================================================
==
Omnibus:                    30186.031   Durbin-Watson:                   1.2
18
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           336709.2
88
Skew:                           1.750   Prob(JB):                         0.
00
Kurtosis:                      13.097   Cond. No.                     2.50e+
17
==============================================================================
==

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
[2] The smallest eigenvalue is 1.23e-30. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

As we can see the clustering effects of volatility still translate into a statistically significant 13-14% 5y autocorrelation as demonstrated by these various regressions.

3. What are the benefits of the panel approach, versus simply running one regression for

each firm? What are the potential costs?

The more variables you add to the regression, the more risk of multi-colinearity (or just generally losing track of causality) you introduce. You also could start having colinearity between the effects of individual explanatory variables, reducing the explanatory power of the individual betas.

The positive effects on the other hand are that these panel regressions can allow the individual beta's of each variable to store exclusively the information related to that variable by assigning the relative noise of external effects to their respective variables. In our case for example, asigning industry and mean-annual effects to panel variables allowed us to extract a more expressive value for the autocorrelation of variances.

It thus comes down to a judgement call when running the regression to find the optimal linearly independent set of risk factors to which the regression will "assign" linearly independent risk-factors in-line with those we are trying to explain.