# Dependencies and Setup  
import pandas as pd  
import scipy.stats as stats  
import statsmodels.api as sm  
import matplotlib.pyplot as plt  
import seaborn as sn  
import numpy as np

# Function for forward select linear model

import statsmodels.formula.api as smf  
  
def forward\_selected(data, response):  
 """Linear model designed by forward selection.  
  
 Parameters:  
 -----------  
 data : pandas DataFrame with all possible predictors and response  
  
 response: string, name of response column in data  
  
 Returns:  
 --------  
 model: an "optimal" fitted statsmodels linear model  
 with an intercept  
 selected by forward selection  
 evaluated by adjusted R-squared  
 """  
 remaining = set(data.columns)  
 remaining.remove(response)  
 selected = []  
 current\_score, best\_new\_score = 0.0, 0.0  
 while remaining and current\_score == best\_new\_score:  
 scores\_with\_candidates = []  
 for candidate in remaining:  
 formula = "{} ~ {}".format(response,  
 ' + '.join(selected + [candidate]))  
 score = smf.ols(formula, data).fit().rsquared\_adj  
 scores\_with\_candidates.append((score, candidate))  
 scores\_with\_candidates.sort()  
 best\_new\_score, best\_candidate = scores\_with\_candidates.pop()  
 if current\_score < best\_new\_score:  
 remaining.remove(best\_candidate)  
 selected.append(best\_candidate)  
 current\_score = best\_new\_score  
 formula = "{} ~ {}".format(response,  
 ' + '.join(selected))  
 model = smf.ols(formula, data).fit()  
 return model

# Importing the csv  
df= pd.read\_csv('fidelity\_mutual\_funds\_return\_w\_risk.csv')  
df.head()

name morningstar\_category \  
0 Baron Partners Fund Institutional Shares (BPTIX) Large Growth   
1 Baron Partners Fund Retail Shares (BPTRX) Large Growth   
2 Morgan Stanley Institutional Fund, Inc. Incept... Small Growth   
3 Morgan Stanley Institutional Fund, Inc. Incept... Small Growth   
4 Morgan Stanley Institutional Fund, Inc. Incept... Small Growth   
  
 ytdDaily yr1 yr3 yr5 yr10 life\_of\_fund net\_expense\_ratio \  
0 44.60 110.27 65.56 47.54 29.02 27.02 1.30   
1 44.28 109.72 65.13 47.15 28.68 20.45 1.56   
2 20.05 81.15 55.11 38.37 22.37 14.14 1.00   
3 19.75 80.67 54.70 37.98 22.01 13.83 1.35   
4 NaN 79.39 53.46 36.90 21.11 13.00 2.10   
  
 gross\_expense\_ratio morningstar\_rating\_overall risk std\_dev \  
0 1.30 1137.0 6 40.41   
1 1.56 1137.0 6 40.39   
2 1.19 574.0 7 40.44   
3 1.45 574.0 7 40.48   
4 2.27 574.0 7 40.42   
  
 sharpe\_ratio\_3\_yr beta r2 minimum\_investment last\_dividend   
0 1.60 1.51 0.63 1000000.0 0.2224   
1 1.59 1.51 0.63 2500.0 0.1243   
2 1.34 1.40 0.71 5000000.0 0.0000   
3 1.32 1.40 0.71 2500.0 0.0000   
4 1.30 1.40 0.71 2500.0 NaN

df.columns

Index(['name', 'morningstar\_category', 'ytdDaily', 'yr1', 'yr3', 'yr5', 'yr10',  
 'life\_of\_fund', 'net\_expense\_ratio', 'gross\_expense\_ratio',  
 'morningstar\_rating\_overall', 'risk', 'std\_dev', 'sharpe\_ratio\_3\_yr',  
 'beta', 'r2', 'minimum\_investment', 'last\_dividend'],  
 dtype='object')

# Feature selection

# Extracting all the numeric columns  
df\_num = df[['ytdDaily', 'yr1', 'yr3', 'yr5', 'yr10',  
 'life\_of\_fund', 'net\_expense\_ratio', 'gross\_expense\_ratio',  
 'morningstar\_rating\_overall', 'risk', 'std\_dev', 'sharpe\_ratio\_3\_yr',  
 'beta', 'r2', 'minimum\_investment', 'last\_dividend']].dropna()  
df\_num

ytdDaily yr1 yr3 yr5 yr10 life\_of\_fund net\_expense\_ratio \  
0 44.60 110.27 65.56 47.54 29.02 27.02 1.30   
1 44.28 109.72 65.13 47.15 28.68 20.45 1.56   
2 20.05 81.15 55.11 38.37 22.37 14.14 1.00   
3 19.75 80.67 54.70 37.98 22.01 13.83 1.35   
5 29.85 78.53 51.07 37.87 21.75 21.12 1.07   
... ... ... ... ... ... ... ...   
9190 50.49 112.78 -11.17 -9.38 -4.58 -1.65 1.35   
9192 37.18 106.34 -16.52 -14.67 -12.41 -3.39 1.42   
9193 36.87 105.81 -16.69 -14.87 -12.72 -3.74 1.68   
9194 36.89 105.82 -16.74 -14.88 -12.62 -3.63 1.68   
9195 36.04 104.30 -17.35 -15.51 -13.28 -4.34 2.43   
  
 gross\_expense\_ratio morningstar\_rating\_overall risk std\_dev \  
0 1.30 1137.0 6 40.41   
1 1.56 1137.0 6 40.39   
2 1.19 574.0 7 40.44   
3 1.45 574.0 7 40.48   
5 1.07 550.0 6 33.96   
... ... ... ... ...   
9190 1.87 72.0 8 48.41   
9192 1.42 72.0 8 56.46   
9193 1.68 72.0 8 56.42   
9194 1.68 72.0 8 56.44   
9195 2.43 72.0 8 56.40   
  
 sharpe\_ratio\_3\_yr beta r2 minimum\_investment last\_dividend   
0 1.60 1.51 0.63 1000000.0 0.222400   
1 1.59 1.51 0.63 2500.0 0.124300   
2 1.34 1.40 0.71 5000000.0 0.000000   
3 1.32 1.40 0.71 2500.0 0.000000   
5 1.47 1.16 0.62 1000000.0 0.002900   
... ... ... ... ... ...   
9190 -0.25 1.12 0.95 2500.0 0.112000   
9192 -0.31 2.44 0.64 2500.0 1.837037   
9193 -0.31 2.44 0.64 2500.0 1.837037   
9194 -0.32 2.44 0.64 2500.0 1.837037   
9195 -0.33 2.44 0.64 2500.0 1.837037   
  
[6203 rows x 16 columns]

# Linear regression model

model = forward\_selected(df\_num.dropna(), 'yr10')  
print("Selected features for the model:")  
print(model.model.formula)  
print("----------------------------------")  
print("Adjusted R squared for the model:")  
print(model.rsquared\_adj)

Selected features for the model:  
yr10 ~ yr5 + yr3 + life\_of\_fund + ytdDaily + yr1 + r2 + gross\_expense\_ratio + morningstar\_rating\_overall + risk + sharpe\_ratio\_3\_yr + std\_dev + net\_expense\_ratio + minimum\_investment  
----------------------------------  
Adjusted R squared for the model:  
0.9444266011578509

print(model.summary())

OLS Regression Results   
==============================================================================  
Dep. Variable: yr10 R-squared: 0.945  
Model: OLS Adj. R-squared: 0.944  
Method: Least Squares F-statistic: 8109.  
Date: Thu, 11 Nov 2021 Prob (F-statistic): 0.00  
Time: 16:48:21 Log-Likelihood: -10247.  
No. Observations: 6203 AIC: 2.052e+04  
Df Residuals: 6189 BIC: 2.062e+04  
Df Model: 13   
Covariance Type: nonrobust   
==============================================================================================  
 coef std err t P>|t| [0.025 0.975]  
----------------------------------------------------------------------------------------------  
Intercept 0.7433 0.105 7.070 0.000 0.537 0.949  
yr5 0.9030 0.011 83.411 0.000 0.882 0.924  
yr3 -0.2693 0.010 -27.497 0.000 -0.288 -0.250  
life\_of\_fund 0.2965 0.008 36.702 0.000 0.281 0.312  
ytdDaily 0.0836 0.005 15.369 0.000 0.073 0.094  
yr1 -0.0261 0.004 -6.132 0.000 -0.035 -0.018  
r2 0.7745 0.069 11.274 0.000 0.640 0.909  
gross\_expense\_ratio -0.2662 0.033 -8.118 0.000 -0.330 -0.202  
morningstar\_rating\_overall 0.0003 5.25e-05 4.770 0.000 0.000 0.000  
risk -0.0632 0.020 -3.179 0.001 -0.102 -0.024  
sharpe\_ratio\_3\_yr -0.5183 0.079 -6.576 0.000 -0.673 -0.364  
std\_dev -0.0300 0.006 -4.701 0.000 -0.043 -0.017  
net\_expense\_ratio 0.1494 0.048 3.093 0.002 0.055 0.244  
minimum\_investment -3.352e-08 2.01e-08 -1.665 0.096 -7.3e-08 5.95e-09  
==============================================================================  
Omnibus: 1135.566 Durbin-Watson: 1.841  
Prob(Omnibus): 0.000 Jarque-Bera (JB): 8786.598  
Skew: -0.662 Prob(JB): 0.00  
Kurtosis: 8.678 Cond. No. 6.40e+06  
==============================================================================  
  
Notes:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.  
[2] The condition number is large, 6.4e+06. This might indicate that there are  
strong multicollinearity or other numerical problems.

# Correlation between the response and explanatory variables

for col in df\_num.columns.difference(['yr10', 'last\_dividend', 'beta']):  
 print(f"Correlation between yr10 and {col} is {stats.pearsonr(df\_num['yr10'].values,df\_num[col].values)[0]}")

Correlation between yr10 and gross\_expense\_ratio is -0.043466501612530156  
Correlation between yr10 and life\_of\_fund is 0.8252743569158877  
Correlation between yr10 and minimum\_investment is 0.03846300222625589  
Correlation between yr10 and morningstar\_rating\_overall is 0.5347603693302172  
Correlation between yr10 and net\_expense\_ratio is 0.0037295280795975154  
Correlation between yr10 and r2 is 0.3601893515895218  
Correlation between yr10 and risk is 0.5510176060654153  
Correlation between yr10 and sharpe\_ratio\_3\_yr is 0.42204222818958054  
Correlation between yr10 and std\_dev is 0.5833578563997679  
Correlation between yr10 and yr1 is 0.6612711971272358  
Correlation between yr10 and yr3 is 0.8749152051615144  
Correlation between yr10 and yr5 is 0.9461939495764184  
Correlation between yr10 and ytdDaily is 0.6567130333247719

# Actual vs predicted plot

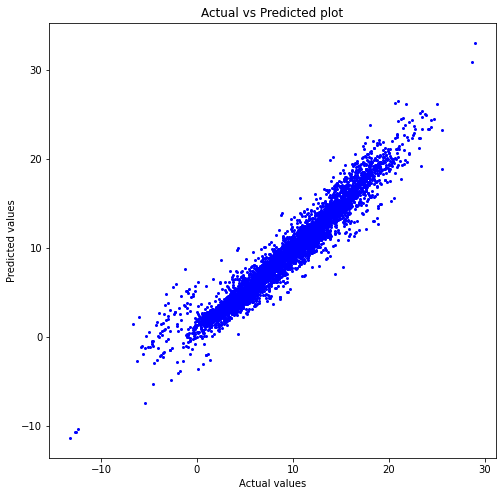
X = df\_num[df\_num.columns.difference(['yr10', 'last\_dividend', 'beta'])].dropna()  
y = df\_num['yr10'].dropna().values  
predictions = model.predict(X).values  
r2 = model.rsquared\_adj

correlation, p\_value = stats.pearsonr(y,predictions)  
print(f"Correlation between actual and predicted is {correlation}")

Correlation between actual and predicted is 0.9718760664044803

plt.figure(figsize=(8,8))  
plt.plot(y, predictions, 'o', color='blue', markersize=2)  
plt.xlabel("Actual values")  
plt.ylabel("Predicted values")  
plt.title('Actual vs Predicted plot')

Text(0.5, 1.0, 'Actual vs Predicted plot')



#### Our model's predictions are pretty close to actual values. Also our R-squared value is 94.5% which is pretty high.

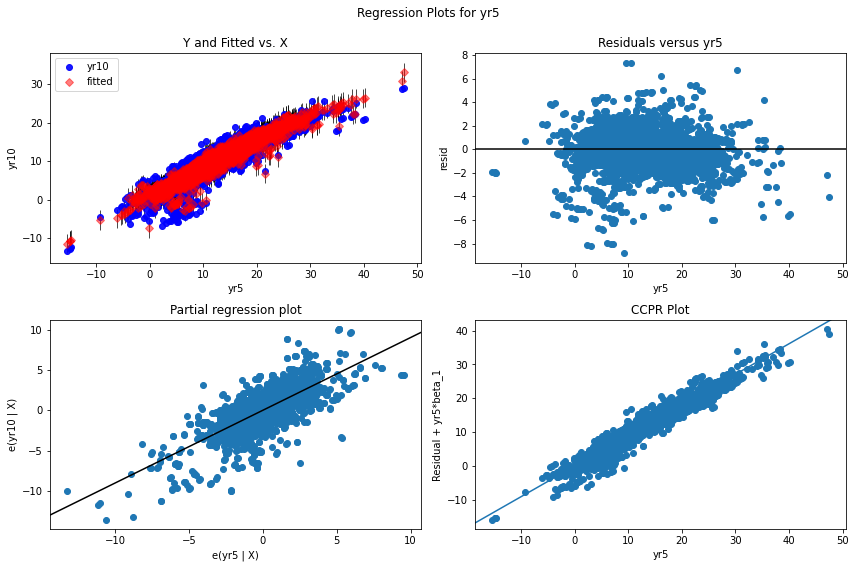
## Model score

from sklearn.linear\_model import LinearRegression  
lin\_reg\_model = LinearRegression()  
  
lin\_reg\_model.fit(X, y)  
print(f"Training Data Score: {lin\_reg\_model.score(X, y)}")

Training Data Score: 0.9445430884498451

# Model plot with one explanatory variable with the highest correlation

#define figure size  
fig = plt.figure(figsize=(12,8))  
  
#produce regression plots  
fig = sm.graphics.plot\_regress\_exog(model, 'yr5', fig=fig)



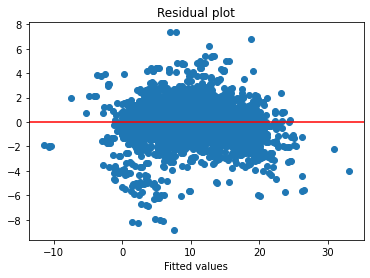
## Residual plot

# Getting residual values  
residuals = y - predictions  
residuals

array([-4.01766782, -2.1700478 , 0.06551326, ..., -2.0163843 ,  
 -1.95942283, -1.89180646])

plt.scatter(predictions, residuals)  
plt.axhline(y = 0, color = 'r', linestyle = '-')  
plt.xlabel("Fitted values")  
plt.title('Residual plot')

Text(0.5, 1.0, 'Residual plot')

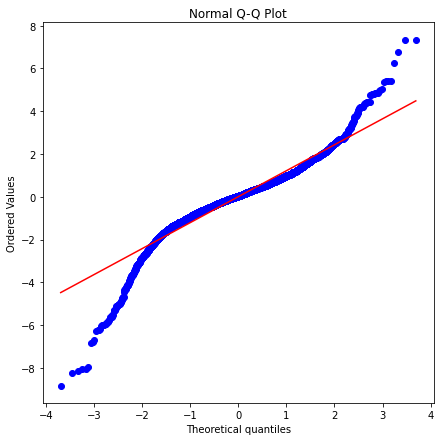


#### From the above plot we observe that the residual against yr10 is slightly curved as most of the data points are below horizontal line. This indicates that a non-linear relation might have given us a better model for this dataset.

## Normal Q-Q Plot

# Plotting residual values on a normall Q-Q plot  
plt.figure(figsize=(7,7))  
stats.probplot(residuals, dist="norm", plot=plt)  
plt.title("Normal Q-Q Plot")

Text(0.5, 1.0, 'Normal Q-Q Plot')



#### From the above plot, we can say that the distribution of residuals is pretty normal.

# Anova of OLS (Ordinary least squared) model

anova\_table = sm.stats.anova\_lm(model)  
anova\_table

df sum\_sq mean\_sq \  
yr5 1.0 159572.166398 159572.166398   
yr3 1.0 4072.959165 4072.959165   
life\_of\_fund 1.0 3199.929739 3199.929739   
ytdDaily 1.0 669.963411 669.963411   
yr1 1.0 365.704202 365.704202   
r2 1.0 188.684904 188.684904   
gross\_expense\_ratio 1.0 111.082308 111.082308   
morningstar\_rating\_overall 1.0 38.323975 38.323975   
risk 1.0 27.740343 27.740343   
sharpe\_ratio\_3\_yr 1.0 50.175736 50.175736   
std\_dev 1.0 32.363664 32.363664   
net\_expense\_ratio 1.0 18.596081 18.596081   
minimum\_investment 1.0 4.427257 4.427257   
Residual 6189.0 9884.449514 1.597100   
  
 F PR(>F)   
yr5 99913.721691 0.000000e+00   
yr3 2550.222371 0.000000e+00   
life\_of\_fund 2003.588073 0.000000e+00   
ytdDaily 419.487554 2.934155e-90   
yr1 228.980208 8.004859e-51   
r2 118.142226 2.842844e-27   
gross\_expense\_ratio 69.552523 9.083572e-17   
morningstar\_rating\_overall 23.995983 9.899106e-07   
risk 17.369200 3.119681e-05   
sharpe\_ratio\_3\_yr 31.416786 2.171539e-08   
std\_dev 20.264023 6.869318e-06   
net\_expense\_ratio 11.643657 6.483453e-04   
minimum\_investment 2.772061 9.597322e-02   
Residual NaN NaN

#### F critical value

# At 5% level of significance   
stats.f.ppf(q=1-.05, dfn=13, dfd=6189)

1.7217356615435946

#### So, from the anova table above, we can conclude that all the explanatory variables have significant variation as their F statistical values are larger than F critical value and they are all statistically significant except minimum\_investment.

# Anova for different years' rates

stats.f\_oneway(df\_num['yr1'], df\_num['yr3'], df\_num['yr5'], df\_num['yr10'])

F\_onewayResult(statistic=2917.3689315577753, pvalue=0.0)

#### As the p-value is close to 0, we can say that the mean returns of each years are different