Name: Erik Alexander Sandvik
Course: STK-IN4300
Assignment number: 2

In [37]:
```
%matplotlib inline

# Import required packages

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Problem 1. Regression

1.

In [38]:
```
# Read data from file

df = pd.read_csv("qsar_aquatic_toxicity.csv", sep=";")


# Label the columns

names = ['TPSA', 'SAacc', 'H050', 'MLOGP', 'RDCHI', 'GATS1p', 'nN', 'C040', 'LC50']
df.columns = names


# Split dataframe by features and response variable

X = df.iloc[:, :-1]
y = df.iloc[:, -1]


# Split the data into a test and training set

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=1)


# Ordinary Least Squares

from sklearn import linear_model
reg = linear_model.LinearRegression()

reg.fit(X_train, y_train)


# Calculate the predicted response using the training set and test set as inputs

y_predict_train = reg.predict(X_train)
y_predict_test = reg.predict(X_test)


# Calculate the training and test error

from sklearn.metrics import mean_squared_error as MSE

print('Training error: {:.2f}'.format(MSE(y_train, y_predict_train)))
print('Test error: {:.2f}'.format(MSE(y_test, y_predict_test)))
```

```
Training error: 1.20
Test error: 1.92
```

In [39]:
```
# Calculate the significance level of each coefficient in the linear model
# scikit-learn doesn't have this implemented

import statsmodels.api as sm
from statsmodels.tools import add_constant

mod = sm.OLS(y_train, add_constant(X_train))
fii = mod.fit()
p_values = fii.summary2().tables[1]['P>|t|']

print("P-values:\n")
print(p_values)
```

```
P-values:

const     6.718007e-19
TPSA      4.488061e-17
SAacc     5.134547e-10
H050      4.744807e-01
MLOGP     4.216208e-12
RDCHI     5.209703e-03
GATS1p    6.349095e-04
nN        5.909546e-03
C040      7.280372e-01
Name: P>|t|, dtype: float64
```

In [40]:
```
# Same thing as above, but with dichotomization of the count variables


# Subscript d for dichotomization

X_d = X.copy()

count_variables = ['H050', 'nN', 'C040']

for cnt_var in count_variables:
    X_d.loc[X[cnt_var] > 0, cnt_var] = 1

X_train_d, X_test_d, y_train, y_test = train_test_split(X_d, y, test_size=0.33, random_state=1)

reg_d = linear_model.LinearRegression()

reg_d.fit(X_train_d, y_train)

y_predict_train_d = reg_d.predict(X_train_d)
y_predict_test_d = reg_d.predict(X_test_d)

print('Training error: {:.2f}'.format(MSE(y_train, y_predict_train_d)))
print('Test error: {:.2f}'.format(MSE(y_test, y_predict_test_d)))

mod = sm.OLS(y_train, add_constant(X_train_d))
fii = mod.fit()
p_values = fii.summary2().tables[1]['P>|t|']

print('\n')
print('P-values:\n')
print(p_values)
```

```
Training error: 1.22
Test error: 2.02


P-values:

const     5.831407e-20
TPSA      4.879928e-13
SAacc     1.190513e-08
H050      1.693085e-01
MLOGP     1.701730e-12
RDCHI     1.398751e-02
GATS1p    4.313739e-04
nN        6.238373e-01
C040      3.955416e-01
Name: P>|t|, dtype: float64
```

With dichotomization the test error increases and the p-values of the coefficients increases by orders of magnitude, with the exception of the variable 'C040' where the p-value of the corresponding coefficient is reduced by a factor of ~2.

2.

In [41]:
```
# Repeat 200 times

av_test_error    = 0
av_test_error_d  = 0

av_test_error2   = 0
av_test_error2_d = 0

n = 200

for i in range(n):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=i)
    X_train_d, X_test_d, y_train, y_test = train_test_split(X_d, y, test_size=0.33, random_state=i)

    reg.fit(X_train, y_train)
    reg_d.fit(X_train_d, y_train)

    y_predict = reg.predict(X_test)
    y_predict_d = reg_d.predict(X_test_d)

    av_test_error += MSE(y_test, y_predict)
    av_test_error_d += MSE(y_test, y_predict_d)

    av_test_error2 += MSE(y_test, y_predict)**2
    av_test_error2_d += MSE(y_test, y_predict_d)**2

av_test_error    /= n
av_test_error_d  /= n
av_test_error2   /= n
av_test_error2_d /= n

std = np.sqrt(av_test_error2 - av_test_error**2)
std_d = np.sqrt(av_test_error2_d - av_test_error_d**2)


print("Average test error: {:.2f}, Standard deviation: {:.2f}".format(av_test_error, std))
print("Average test error dichotomized: {:.2f}, Standard deviation: {:.2f}".format(av_test_error_d, s
td_d))
```

```
Average test error: 1.48, Standard deviation: 0.17
Average test error dichotomized: 1.53, Standard deviation: 0.17
```

The average test errors are lower than what was obtained before (1.92 and 2.02), but that was just for one particular way of splitting the data for training and testing, where I suppose I just got very unlucky (see the standard deviation).

My best guess for why dichotomization leads to a worse result is that dichotomization makes the linear model biased. But the variance of the prediction is about the same with or without dichotomization. According to the bias-variance decomposition, the expected prediction error is increased with dichotomization.

3.

In [42]:
```
# The first training/test split again

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=1)
reg.fit(X_train, y_train)

y_predict_test = reg.predict(X_test)

error = MSE(y_test, y_predict_test)

coeffs = reg.coef_

print("Error: {:.2f}".format(error))
print("Coefficients: ", np.around(coeffs, decimals=2))
```

```
Error: 1.92
Coefficients:  [ 0.03 -0.02  0.05  0.52  0.43 -0.62 -0.15 -0.04]
```

In [43]:
```
# Automated Stepwise forward and backward selection in python
# See https://github.com/talhahascelik/python_stepwiseSelection for details

import stepwiseSelection as ss

# Backward selection with AIC
final_vars, _ = ss.backwardSelection(X_train, y_train, model_type ="linear", elimination_criteria =
"aic")
```

```
Character Variables (Dummies Generated, First Dummies Dropped): []
Eliminated : C040
Eliminated : H050
                            OLS Regression Results
==============================================================================
Dep. Variable:                   LC50   R-squared:                       0.550
Model:                            OLS   Adj. R-squared:                  0.543
Method:                 Least Squares   F-statistic:                     72.95
Date:                Tue, 19 Oct 2021   Prob (F-statistic):           4.03e-59
Time:                        12:02:17   Log-Likelihood:                -552.21
No. Observations:                 365   AIC:                             1118.
Df Residuals:                     358   BIC:                             1146.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      2.7475      0.270     10.191      0.000       2.217       3.278
TPSA           0.0275      0.003      8.901      0.000       0.021       0.034
SAacc         -0.0144      0.002     -7.495      0.000      -0.018      -0.011
MLOGP          0.5003      0.068      7.411      0.000       0.368       0.633
RDCHI          0.4330      0.152      2.850      0.005       0.134       0.732
GATS1p        -0.6624      0.172     -3.844      0.000      -1.001      -0.324
nN            -0.1422      0.053     -2.687      0.008      -0.246      -0.038
==============================================================================
Omnibus:                       31.890   Durbin-Watson:                   1.768
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               40.429
Skew:                           0.674   Prob(JB):                     1.66e-09
Kurtosis:                       3.917   Cond. No.                         581.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
AIC: 1118.4177729055477
BIC: 1145.717054380625
Final Variables: ['intercept', 'TPSA', 'SAacc', 'MLOGP', 'RDCHI', 'GATS1p', 'nN']
```

Please excuse all the junk.

We see that the features 'C040' and 'H050' were eliminated.

In [44]:
```
X_fs_aic = X.copy()

X_fs_aic.drop('C040', axis=1, inplace=True)
X_fs_aic.drop('H050', axis=1, inplace=True)

X_train_fs_aic, X_test_fs_aic, y_train_fs_aic, y_test_fs_aic = train_test_split(X_fs_aic, y, test_siz
e=0.33, random_state=1)

reg.fit(X_train_fs_aic, y_train_fs_aic)

y_predict_test = reg.predict(X_test_fs_aic)

error = MSE(y_test, y_predict_test)

coeffs = reg.coef_

print("Error: {:.2f}".format(error))
print("Coefficients: ", np.around(coeffs, decimals=2))
```

```
Error: 1.92
Coefficients:  [ 0.03 -0.01  0.5   0.43 -0.66 -0.14]
```

In [45]:
```
# Backward selection with BIC
final_vars, _ = ss.backwardSelection(X_train, y_train, model_type ="linear", elimination_criteria =
"bic")
```

```
Character Variables (Dummies Generated, First Dummies Dropped): []
Eliminated : C040
Eliminated : H050
                            OLS Regression Results
==============================================================================
Dep. Variable:                   LC50   R-squared:                       0.550
Model:                            OLS   Adj. R-squared:                  0.543
Method:                 Least Squares   F-statistic:                     72.95
Date:                Tue, 19 Oct 2021   Prob (F-statistic):           4.03e-59
Time:                        12:02:17   Log-Likelihood:                -552.21
No. Observations:                 365   AIC:                             1118.
Df Residuals:                     358   BIC:                             1146.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      2.7475      0.270     10.191      0.000       2.217       3.278
TPSA           0.0275      0.003      8.901      0.000       0.021       0.034
SAacc         -0.0144      0.002     -7.495      0.000      -0.018      -0.011
MLOGP          0.5003      0.068      7.411      0.000       0.368       0.633
RDCHI          0.4330      0.152      2.850      0.005       0.134       0.732
GATS1p        -0.6624      0.172     -3.844      0.000      -1.001      -0.324
nN            -0.1422      0.053     -2.687      0.008      -0.246      -0.038
==============================================================================
Omnibus:                       31.890   Durbin-Watson:                   1.768
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               40.429
Skew:                           0.674   Prob(JB):                     1.66e-09
Kurtosis:                       3.917   Cond. No.                         581.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
AIC: 1118.4177729055477
BIC: 1145.717054380625
Final Variables: ['intercept', 'TPSA', 'SAacc', 'MLOGP', 'RDCHI', 'GATS1p', 'nN']
```

The same couple of variables were eliminated, so the error and coefficients are the same with backward selection for both AIC and BIC.

In [46]:
```
# Forward selection with AIC
final_vars, _ = ss.forwardSelection(X_train, y_train, model_type ="linear", elimination_criteria = "a
ic")
```

```
Character Variables (Dummies Generated, First Dummies Dropped): []
Entered : MLOGP         AIC : 1237.49891950763
Entered : TPSA  AIC : 1179.6284568355122
Entered : SAacc         AIC : 1134.7105633054348
Entered : GATS1p        AIC : 1128.0436044458704
Entered : RDCHI         AIC : 1123.7030733885122
Entered : nN    AIC : 1118.4177729055477
Break : Significance Level
                            OLS Regression Results
==============================================================================
Dep. Variable:                   LC50   R-squared:                       0.550
Model:                            OLS   Adj. R-squared:                  0.543
Method:                 Least Squares   F-statistic:                     72.95
Date:                Tue, 19 Oct 2021   Prob (F-statistic):           4.03e-59
Time:                        12:02:17   Log-Likelihood:                -552.21
No. Observations:                 365   AIC:                             1118.
Df Residuals:                     358   BIC:                             1146.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      2.7475      0.270     10.191      0.000       2.217       3.278
MLOGP          0.5003      0.068      7.411      0.000       0.368       0.633
TPSA           0.0275      0.003      8.901      0.000       0.021       0.034
SAacc         -0.0144      0.002     -7.495      0.000      -0.018      -0.011
GATS1p        -0.6624      0.172     -3.844      0.000      -1.001      -0.324
RDCHI          0.4330      0.152      2.850      0.005       0.134       0.732
nN            -0.1422      0.053     -2.687      0.008      -0.246      -0.038
==============================================================================
Omnibus:                       31.890   Durbin-Watson:                   1.768
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               40.429
Skew:                           0.674   Prob(JB):                     1.66e-09
Kurtosis:                       3.917   Cond. No.                         581.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
AIC: 1118.4177729055477
BIC: 1145.717054380625
Final Variables: ['intercept', 'MLOGP', 'TPSA', 'SAacc', 'GATS1p', 'RDCHI', 'nN']
```

H050 and C040 were eliminated yet again.

In [47]:
```
# Forward selection with BIC
final_vars, _ = ss.forwardSelection(X_train, y_train, model_type ="linear", elimination_criteria = "b
ic")
```

```
Character Variables (Dummies Generated, First Dummies Dropped): []
Entered : MLOGP         BIC : 1245.298714214795
Entered : TPSA  BIC : 1191.3281488962598
Entered : SAacc         BIC : 1150.3101527197648
Entered : GATS1p        BIC : 1147.5430912137829
Entered : RDCHI         BIC : 1147.102457510007
Entered : nN    BIC : 1145.717054380625
Break : Significance Level
                            OLS Regression Results
==============================================================================
Dep. Variable:                   LC50   R-squared:                       0.550
Model:                            OLS   Adj. R-squared:                  0.543
Method:                 Least Squares   F-statistic:                     72.95
Date:                Tue, 19 Oct 2021   Prob (F-statistic):           4.03e-59
Time:                        12:02:18   Log-Likelihood:                -552.21
No. Observations:                 365   AIC:                             1118.
Df Residuals:                     358   BIC:                             1146.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      2.7475      0.270     10.191      0.000       2.217       3.278
MLOGP          0.5003      0.068      7.411      0.000       0.368       0.633
TPSA           0.0275      0.003      8.901      0.000       0.021       0.034
SAacc         -0.0144      0.002     -7.495      0.000      -0.018      -0.011
GATS1p        -0.6624      0.172     -3.844      0.000      -1.001      -0.324
RDCHI          0.4330      0.152      2.850      0.005       0.134       0.732
nN            -0.1422      0.053     -2.687      0.008      -0.246      -0.038
==============================================================================
Omnibus:                       31.890   Durbin-Watson:                   1.768
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               40.429
Skew:                           0.674   Prob(JB):                     1.66e-09
Kurtosis:                       3.917   Cond. No.                         581.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
AIC: 1118.4177729055477
BIC: 1145.717054380625
Final Variables: ['intercept', 'MLOGP', 'TPSA', 'SAacc', 'GATS1p', 'RDCHI', 'nN']
```

So both backward and forward selection, with AIC or with BIC, leads to the exclusion of the variables 'H050' and 'C040', which are the third and the last feature respectively. With or without exclusion, the prediction error is about the same.

For your convenience (so you don't have to scroll through all the junk), the coefficients in the linear model without exclusion are

[0.03 -0.02 0.05 0.52 0.43 -0.62 -0.15 -0.04]

With exclusion, the coefficients in the linear model are

[0.03 -0.01 0.5 0.43 -0.66 -0.14]

The coefficients of the features that are kept have nearly unchanged values, only differing at most in the second decimal place.

4.

In [57]:
```
from sklearn.linear_model import RidgeCV
from sklearn.metrics import make_scorer

N, p = X.shape


def deviance(y_true, y_pred):
    """
    deviance = -2*log(p(y_true | estimated parameters))
    """
    sum_y_true_minus_y_pred_squared = np.sum((y_true - y_pred)**2)

    sigma2 = sum_y_true_minus_y_pred_squared/(N - p - 1)

    log_prob_y_true = -sum_y_true_minus_y_pred_squared/(2*sigma2)

    return -2*log_prob_y_true # unnormalized but doesn't matter


score = make_scorer(deviance, greater_is_better=False)

reg = RidgeCV(alphas=np.logspace(-20, -11, 200), cv=10, scoring=score)
reg.fit(X_train, y_train)


print('Complexity parameter: {}'.format(reg.alpha_))
print('Coefficients:\n', np.around(reg.coef_, decimals=2))

print('Test error: {:.2f}'.format(MSE(y_test, reg.predict(X_test))))
```

```
Complexity parameter: 1e-20
Coefficients:
 [ 0.03 -0.02  0.05  0.52  0.43 -0.62 -0.15 -0.04]
Test error: 1.92
```

So I get the smallest considered value of the complexity parameter no matter what grid is specified like the problem text suggests. No idea why though.

No plot is provided because it requires messing around with the deep inner-workings of scikit-learn.

Aaaaaaaand that's how far I got with this assignment ....