Experiment No.8

Title: Attribute subset selection

Batch:B4 **Roll No.:16010420133** **Experiment No.: 8**

Aim: Attribute subset selection

Resources needed: Python

Results:

FDS EXP 8 - Jupyter Notebook

localhost:8889/notebooks/FDS%20EXP%208.ipynb

jupyter FDS EXP 8 Last Checkpoint: 34 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [10]:
```
%matplotlib inline
import pandas as pd
import numpy as np
import itertools
import time
import statsmodels.api as sm
import matplotlib.pyplot as plt
```

Best Subset Selection

In [12]:
```
hitters_df = pd.read_csv('Hitters1.csv')
hitters_df.head()
```

Out[12]:

	Unnamed: 0	AtBat	Hits	HmRun	Runs	RBI	Walks	Years	CAtBat	CHits	...	CRuns	CRBI	CWalks	League	Division	PutOuts	Assists	Errors	Salary	N
0	-Andy Allanson	293	66	1	30	29	14	1	293	66	...	30	29	14	A	E	446	33	20	NaN	
1	-Alan Ashby	315	81	7	24	38	39	14	3449	835	...	321	414	375	N	W	632	43	10	475.0	
2	-Alvin Davis	479	130	18	66	72	76	3	1624	457	...	224	266	263	A	W	880	82	14	480.0	
3	-Andre Dawson	496	141	20	65	78	37	11	5628	1575	...	828	838	354	N	E	200	11	3	500.0	
4	-Andres Galarraga	321	87	10	39	42	30	2	396	101	...	48	46	33	N	E	805	40	4	91.5	

5 rows x 21 columns

FDS EXP 8 - Jupyter Notebook | screen snip shortcut key - Search

localhost:8889/notebooks/FDS%20EXP%208.ipynb

jupyter FDS EXP 8 Last Checkpoint: 36 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [13]:
```
print("Number of null values:", hitters_df["Salary"].isnull().sum())
```

```
Number of null values: 59
```

In [15]:
```
# Print the dimensions of the original Hitters data (322 rows x 20 columns)
print("Dimensions of original data:", hitters_df.shape)

# Drop any rows the contain missing values, along with the player names
hitters_df_clean = hitters_df.dropna().drop('Unnamed: 0', axis=1)

# Print the dimensions of the modified Hitters data (263 rows x 20 columns)
print("Dimensions of modified data:", hitters_df_clean.shape)

# One last check: should return 0
print("Number of null values:", hitters_df_clean["Salary"].isnull().sum())
```

```
Dimensions of original data: (322, 21)
Dimensions of modified data: (263, 20)
Number of null values: 0
```

In [16]:
```
dummies = pd.get_dummies(hitters_df_clean[['League', 'Division', 'NewLeague']])

y = hitters_df_clean.Salary

# Drop the column with the independent variable (Salary), and columns for which we created dummy variables
X_ = hitters_df_clean.drop(['Salary', 'League', 'Division', 'NewLeague'], axis=1).astype('float64')

# Define the feature set X.
X = pd.concat([X_, dummies[['League_N', 'Division_W', 'NewLeague_N']]], axis=1)
```

In [18]:
```
def processSubset(feature_set):
```

FDS EXP 8 Last Checkpoint: 36 minutes ago (autosaved)

```
In [18]: def processSubset(feature_set):
             model = sm.OLS(y,X[list(feature_set)])
             regr = model.fit()
             RSS = ((regr.predict(X[list(feature_set)]) - y) ** 2).sum()
             return {"model":regr, "RSS":RSS}
```

```
In [20]: def getBest(k):

             tic = time.time()

             results = []

             for combo in itertools.combinations(X.columns, k):
                 results.append(processSubset(combo))

             models = pd.DataFrame(results)

             best_model = models.loc[models['RSS'].argmin()]

             toc = time.time()
             print("Processed", models.shape[0], "models on", k, "predictors in", (toc-tic), "seconds.")
             return best_model
```

```
In [21]: models_best = pd.DataFrame(columns=["RSS", "model"])

         tic = time.time()
         for i in range(1,8):
             models_best.loc[i] = getBest(i)

         toc = time.time()
         print("Total elapsed time:", (toc-tic), "seconds.")
```

```
Processed 19 models on 1 predictors in 0.8721117973327637 seconds.
```

FDS EXP 8 Last Checkpoint: 36 minutes ago (autosaved)

```
Processed 19 models on 1 predictors in 0.8721117973327637 seconds.
Processed 171 models on 2 predictors in 0.9056549072265625 seconds.
Processed 969 models on 3 predictors in 4.294354438781738 seconds.
Processed 3876 models on 4 predictors in 19.90070104598999 seconds.
Processed 11628 models on 5 predictors in 60.4704532623291 seconds.
Processed 27132 models on 6 predictors in 135.66139197349548 seconds.
Processed 50388 models on 7 predictors in 248.69121956825256 seconds.
Total elapsed time: 487.53076100349426 seconds.
```

```
In [22]: models_best
```

Out[22]:

	RSS	model
1	4.321393e+07	<statsmodels.regression.linear_model.Regressio...
2	3.073305e+07	<statsmodels.regression.linear_model.Regressio...
3	2.941071e+07	<statsmodels.regression.linear_model.Regressio...
4	2.797678e+07	<statsmodels.regression.linear_model.Regressio...
5	2.718780e+07	<statsmodels.regression.linear_model.Regressio...
6	2.639772e+07	<statsmodels.regression.linear_model.Regressio...
7	2.606413e+07	<statsmodels.regression.linear_model.Regressio...

```
In [23]: print(models_best.loc[2, "model"].summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                 Salary   R-squared (uncentered):                   0.761
Model:                            OLS   Adj. R-squared (uncentered):              0.760
Method:                 Least Squares   F-statistic:                              416.7
Date:                Fri, 29 Apr 2022   Prob (F-statistic):                    5.80e-82
Time:                        16:22:48   Log-Likelihood:                         -1907.6
```

```
                            OLS Regression Results
=======================================================================================
Dep. Variable:                 Salary   R-squared (uncentered):                   0.761
Model:                            OLS   Adj. R-squared (uncentered):              0.760
Method:                 Least Squares   F-statistic:                              416.7
Date:                Fri, 29 Apr 2022   Prob (F-statistic):                    5.80e-82
Time:                        16:22:48   Log-Likelihood:                         -1907.6
No. Observations:                 263   AIC:                                      3819.
Df Residuals:                     261   BIC:                                      3826.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Hits           2.9538      0.261     11.335      0.000       2.441       3.467
CRBI           0.6788      0.066     10.295      0.000       0.549       0.809
==============================================================================
Omnibus:                      117.551   Durbin-Watson:                   1.933
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              654.612
Skew:                           1.729   Prob(JB):                     7.12e-143
Kurtosis:                       9.912   Cond. No.                         5.88
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [24]:

```
print(getBest(19)["model"].summary())
```

```
Processed 1 models on 19 predictors in 0.4116859436035156 seconds.
                            OLS Regression Results
=======================================================================================
Dep. Variable:                 Salary   R-squared (uncentered):                   0.810
Model:                            OLS   Adj. R-squared (uncentered):              0.795
```

```
Date:                Fri, 29 Apr 2022   Prob (F-statistic):                    1.31e-76
Time:                        16:22:49   Log-Likelihood:                         -1877.9
No. Observations:                 263   AIC:                                      3794.
Df Residuals:                     244   BIC:                                      3862.
Df Model:                          19
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
AtBat          -1.5975      0.600     -2.663      0.008      -2.779      -0.416
Hits            7.0330      2.374      2.963      0.003       2.357      11.709
HmRun           4.1210      6.229      0.662      0.509      -8.148      16.390
Runs           -2.3776      2.994     -0.794      0.428      -8.276       3.520
RBI            -1.0873      2.613     -0.416      0.678      -6.234       4.059
Walks           6.1560      1.836      3.352      0.001       2.539       9.773
Years           9.5196     10.128      0.940      0.348     -10.429      29.468
CAtBat         -0.2018      0.135     -1.497      0.136      -0.467       0.064
CHits           0.1380      0.678      0.204      0.839      -1.197       1.473
CHmRun         -0.1669      1.625     -0.103      0.918      -3.367       3.033
CRuns           1.5070      0.753      2.001      0.047       0.023       2.991
CRBI            0.7742      0.696      1.113      0.267      -0.596       2.144
CWalks         -0.7851      0.329     -2.384      0.018      -1.434      -0.137
PutOuts         0.2856      0.078      3.673      0.000       0.132       0.439
Assists         0.3137      0.220      1.427      0.155      -0.119       0.747
Errors         -2.0463      4.350     -0.470      0.638     -10.615       6.522
League_N       86.8139     78.463      1.106      0.270     -67.737     241.365
Division_W    -97.5160     39.084     -2.495      0.013    -174.500     -20.532
NewLeague_N   -23.9133     79.361     -0.301      0.763    -180.234     132.407
==============================================================================
Omnibus:                       97.217   Durbin-Watson:                   2.024
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              626.205
Skew:                           1.320   Prob(JB):                     1.05e-136
Kurtosis:                      10.083   Cond. No.                     2.06e+04
==============================================================================
```

FDS EXP 8 - Jupyter Notebook

localhost:8889/notebooks/FDS%20EXP%208.ipynb

jupyter FDS EXP 8 Last Checkpoint: 37 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 2.06e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [25]:
```
models_best.loc[2, "model"].rsquared
```

Out[25]:
```
0.7614950002332872
```

In [26]:
```
models_best.apply(lambda row: row[1].rsquared, axis=1)
```

Out[26]:
```
1    0.664637
2    0.761495
3    0.771757
4    0.782885
5    0.789008
6    0.795140
7    0.797728
dtype: float64
```

In [27]:
```
plt.figure(figsize=(20,10))
plt.rcParams.update({'font.size': 18, 'lines.markersize': 10})

plt.subplot(2, 2, 1)

plt.plot(models_best["RSS"])
plt.xlabel('# Predictors')
plt.ylabel('RSS')

rsquared_adj = models_best.apply(lambda row: row[1].rsquared_adj, axis=1)

plt.subplot(2, 2, 2)
```

FDS EXP 8 - Jupyter Notebook

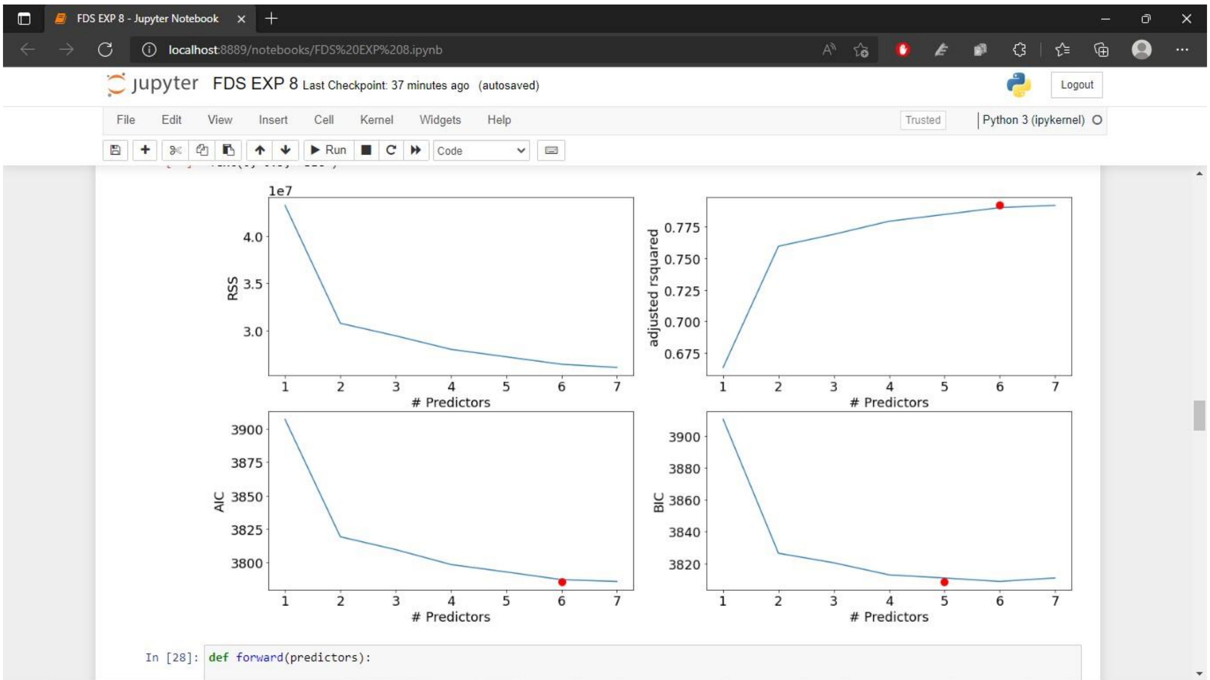
localhost:8889/notebooks/FDS%20EXP%208.ipynb

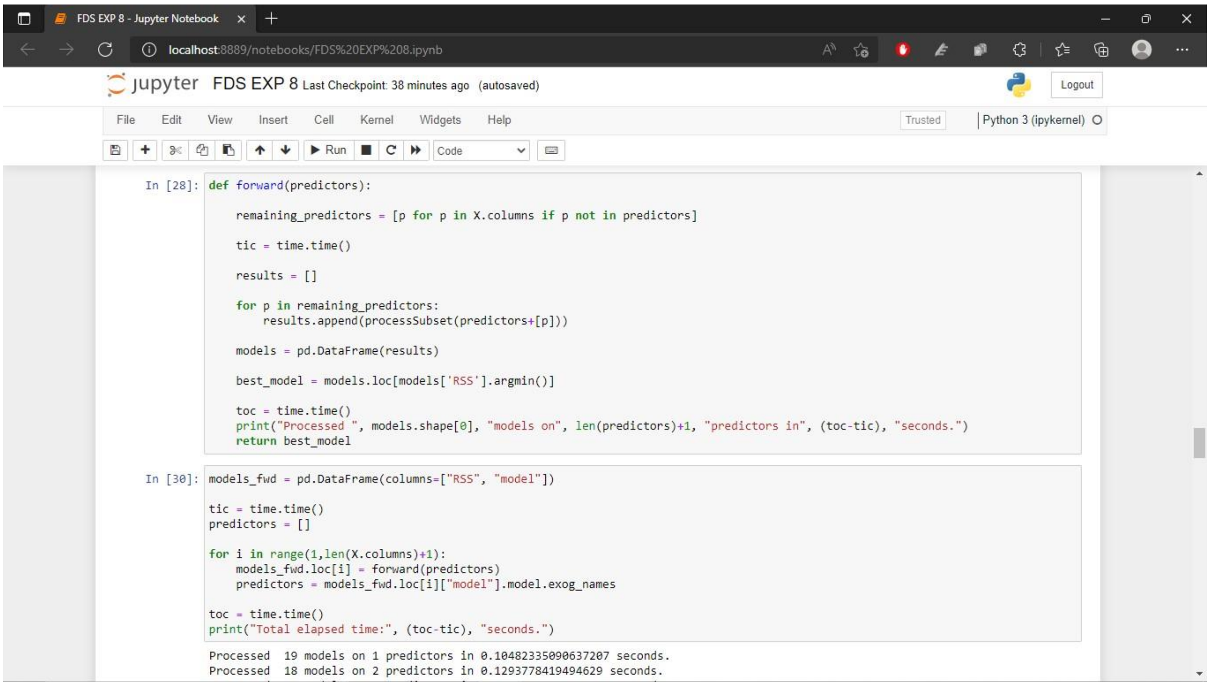
jupyter FDS EXP 8 Last Checkpoint: 37 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [27]:
```
plt.figure(figsize=(20,10))
plt.rcParams.update({'font.size': 18, 'lines.markersize': 10})

plt.subplot(2, 2, 1)

plt.plot(models_best["RSS"])
plt.xlabel('# Predictors')
plt.ylabel('RSS')

rsquared_adj = models_best.apply(lambda row: row[1].rsquared_adj, axis=1)

plt.subplot(2, 2, 2)
plt.plot(rsquared_adj)
plt.plot(rsquared_adj.argmax(), rsquared_adj.max(), "or")
plt.xlabel('# Predictors')
plt.ylabel('adjusted rsquared')

aic = models_best.apply(lambda row: row[1].aic, axis=1)

plt.subplot(2, 2, 3)
plt.plot(aic)
plt.plot(aic.argmin(), aic.min(), "or")
plt.xlabel('# Predictors')
plt.ylabel('AIC')

bic = models_best.apply(lambda row: row[1].bic, axis=1)

plt.subplot(2, 2, 4)
plt.plot(bic)
plt.plot(bic.argmin(), bic.min(), "or")
plt.xlabel('# Predictors')
plt.ylabel('BIC')
```

Out[27]:
```
Text(0, 0.5, 'BIC')
```





```
In [28]: def forward(predictors):

             remaining_predictors = [p for p in X.columns if p not in predictors]

             tic = time.time()

             results = []

             for p in remaining_predictors:
                 results.append(processSubset(predictors+[p]))

             models = pd.DataFrame(results)

             best_model = models.loc[models['RSS'].argmin()]

             toc = time.time()
             print("Processed ", models.shape[0], "models on", len(predictors)+1, "predictors in", (toc-tic), "seconds.")
             return best_model

In [30]: models_fwd = pd.DataFrame(columns=["RSS", "model"])

         tic = time.time()
         predictors = []

         for i in range(1,len(X.columns)+1):
             models_fwd.loc[i] = forward(predictors)
             predictors = models_fwd.loc[i]["model"].model.exog_names

         toc = time.time()
         print("Total elapsed time:", (toc-tic), "seconds.")

         Processed  19 models on 1 predictors in 0.10482335090637207 seconds.
         Processed  18 models on 2 predictors in 0.1293778419494629 seconds.
```

```
toc = time.time()
print("Total elapsed time:", (toc-tic), "seconds.")
```

```
Processed  19 models on 1 predictors in 0.10482335090637207 seconds.
Processed  18 models on 2 predictors in 0.1293778419494629 seconds.
Processed  17 models on 3 predictors in 0.06249642372131348 seconds.
Processed  16 models on 4 predictors in 0.08460760116577148 seconds.
Processed  15 models on 5 predictors in 0.09275126457214355 seconds.
Processed  14 models on 6 predictors in 0.07052493095397949 seconds.
Processed  13 models on 7 predictors in 0.07083439826965332 seconds.
Processed  12 models on 8 predictors in 0.06045079231262207 seconds.
Processed  11 models on 9 predictors in 0.08232378959655762 seconds.
Processed  10 models on 10 predictors in 0.05585050582885742 seconds.
Processed  9 models on 11 predictors in 0.05361151695251465 seconds.
Processed  8 models on 12 predictors in 0.0403139591217041 seconds.
Processed  7 models on 13 predictors in 0.0483705997467041 seconds.
Processed  6 models on 14 predictors in 0.0540008544921875 seconds.
Processed  5 models on 15 predictors in 0.04001474380493164 seconds.
Processed  4 models on 16 predictors in 0.046875715255737305 seconds.
Processed  3 models on 17 predictors in 0.029920101165771484 seconds.
Processed  2 models on 18 predictors in 0.014958381652832031 seconds.
Processed  1 models on 19 predictors in 0.010971307754516602 seconds.
Total elapsed time: 1.266430377960205 seconds.
```

In [31]:
```
print(models_fwd.loc[1, "model"].summary())
print(models_fwd.loc[2, "model"].summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                 Salary   R-squared (uncentered):                   0.665
Model:                            OLS   Adj. R-squared (uncentered):              0.663
Method:                 Least Squares   F-statistic:                              519.2
Date:                Fri, 29 Apr 2022   Prob (F-statistic):                    4.20e-64
Time:                        16:26:48   Log-Likelihood:                         -1952.4
```

```
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Hits           4.8833      0.214     22.787      0.000       4.461       5.305
==============================================================================
Omnibus:                       90.075   Durbin-Watson:                   1.949
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              293.080
Skew:                           1.469   Prob(JB):                     2.28e-64
Kurtosis:                       7.256   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                 Salary   R-squared (uncentered):                   0.761
Model:                            OLS   Adj. R-squared (uncentered):              0.760
Method:                 Least Squares   F-statistic:                              416.7
Date:                Fri, 29 Apr 2022   Prob (F-statistic):                    5.80e-82
Time:                        16:26:48   Log-Likelihood:                         -1907.6
No. Observations:                 263   AIC:                                      3819.
Df Residuals:                     261   BIC:                                      3826.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Hits           2.9538      0.261     11.335      0.000       2.441       3.467
CRBI           0.6788      0.066     10.295      0.000       0.549       0.809
==============================================================================
Omnibus:                      117.551   Durbin-Watson:                   1.933
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              654.612
Skew:                           1.729   Prob(JB):                    7.12e-143
Kurtosis:                       9.912   Cond. No.                         5.88
```

FDS EXP 8 - Jupyter Notebook

localhost:8889/notebooks/FDS%20EXP%208.ipynb

jupyter FDS EXP 8 Last Checkpoint: 38 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

```
Kurtosis:                       9.912   Cond. No.                         5.88
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [32]:

```python
print(models_best.loc[6, "model"].summary())
print(models_fwd.loc[6, "model"].summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                 Salary   R-squared (uncentered):                   0.795
Model:                            OLS   Adj. R-squared (uncentered):              0.790
Method:                 Least Squares   F-statistic:                              166.3
Date:                Fri, 29 Apr 2022   Prob (F-statistic):                    1.79e-85
Time:                        16:27:02   Log-Likelihood:                         -1887.6
No. Observations:                 263   AIC:                                      3787.
Df Residuals:                     257   BIC:                                      3809.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
AtBat         -1.5488      0.477     -3.248      0.001      -2.488      -0.610
Hits           7.0190      1.613      4.352      0.000       3.843      10.195
Walks          3.7513      1.212      3.095      0.002       1.364       6.138
CRBI           0.6544      0.064     10.218      0.000       0.528       0.781
PutOuts        0.2703      0.075      3.614      0.000       0.123       0.418
```

Backward Selection

FDS EXP 8 - Jupyter Notebook

localhost:8889/notebooks/FDS%20EXP%208.ipynb

jupyter FDS EXP 8 Last Checkpoint: 39 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

Backward Selection

In [34]:

```python
def backward(predictors):

    tic = time.time()

    results = []

    for combo in itertools.combinations(predictors, len(predictors)-1):
        results.append(processSubset(combo))

    # Wrap everything up in a nice dataframe
    models = pd.DataFrame(results)

    # Choose the model with the highest RSS
    best_model = models.loc[models['RSS'].argmin()]

    toc = time.time()
    print("Processed ", models.shape[0], "models on", len(predictors)-1, "predictors in", (toc-tic), "seconds.")

    # Return the best model, along with some other useful information about the model
    return best_model
models_bwd = pd.DataFrame(columns=["RSS", "model"], index = range(1,len(X.columns)))

tic = time.time()
predictors = X.columns

while(len(predictors) > 1):
    models_bwd.loc[len(predictors)-1] = backward(predictors)
    predictors = models_bwd.loc[len(predictors)-1]["model"].model.exog_names

toc = time.time()
print("Total elapsed time:", (toc-tic), "seconds.")
```

```
Processed  19 models on 18 predictors in 0.17781400680541992 seconds.
Processed  18 models on 17 predictors in 0.10025596618652344 seconds.
Processed  17 models on 16 predictors in 0.08572101593017578 seconds.
Processed  16 models on 15 predictors in 0.09221625328063965 seconds.
Processed  15 models on 14 predictors in 0.08062934875488281 seconds.
Processed  14 models on 13 predictors in 0.10401725769042969 seconds.
Processed  13 models on 12 predictors in 0.09275007247924805 seconds.
Processed  12 models on 11 predictors in 0.06920146942138672 seconds.
Processed  11 models on 10 predictors in 0.08178162574768066 seconds.
Processed  10 models on 9 predictors in 0.06881570816040039 seconds.
Processed  9 models on 8 predictors in 0.06482625007629395 seconds.
Processed  8 models on 7 predictors in 0.05983924865722656 seconds.
Processed  7 models on 6 predictors in 0.04188942909240723 seconds.
Processed  6 models on 5 predictors in 0.0312039852142334 seconds.
Processed  5 models on 4 predictors in 0.020160436630249023 seconds.
Processed  4 models on 3 predictors in 0.022186756134033203 seconds.
Processed  3 models on 2 predictors in 0.018112897872924805 seconds.
Processed  2 models on 1 predictors in 0.010102272033691406 seconds.
Total elapsed time: 1.2546279430389404 seconds.
```

```
In [36]: models_bwd = pd.DataFrame(columns=["RSS", "model"], index = range(1,len(X.columns)))

tic = time.time()
predictors = X.columns

while(len(predictors) > 1):
    models_bwd.loc[len(predictors)-1] = backward(predictors)
    predictors = models_bwd.loc[len(predictors)-1]["model"].model.exog_names

toc = time.time()
print("Total elapsed time:", (toc-tic), "seconds.")
```

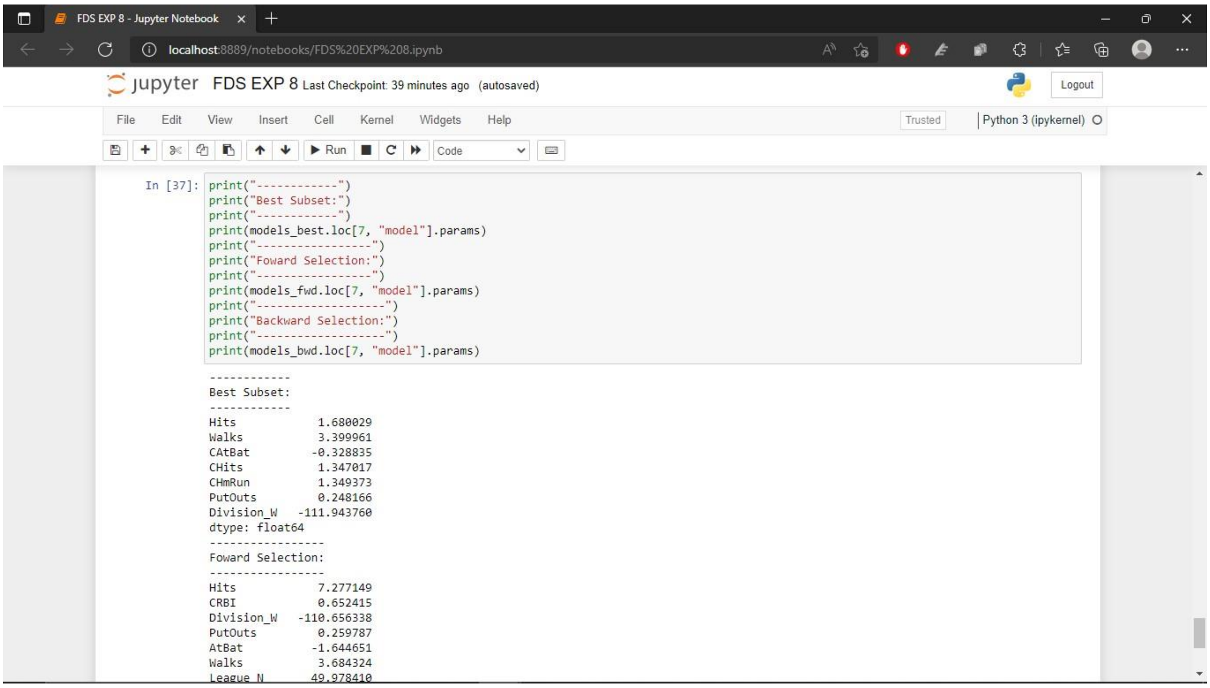
```
Processed  19 models on 18 predictors in 0.14936232566833496 seconds.
Processed  18 models on 17 predictors in 0.1008610725402832 seconds.
```

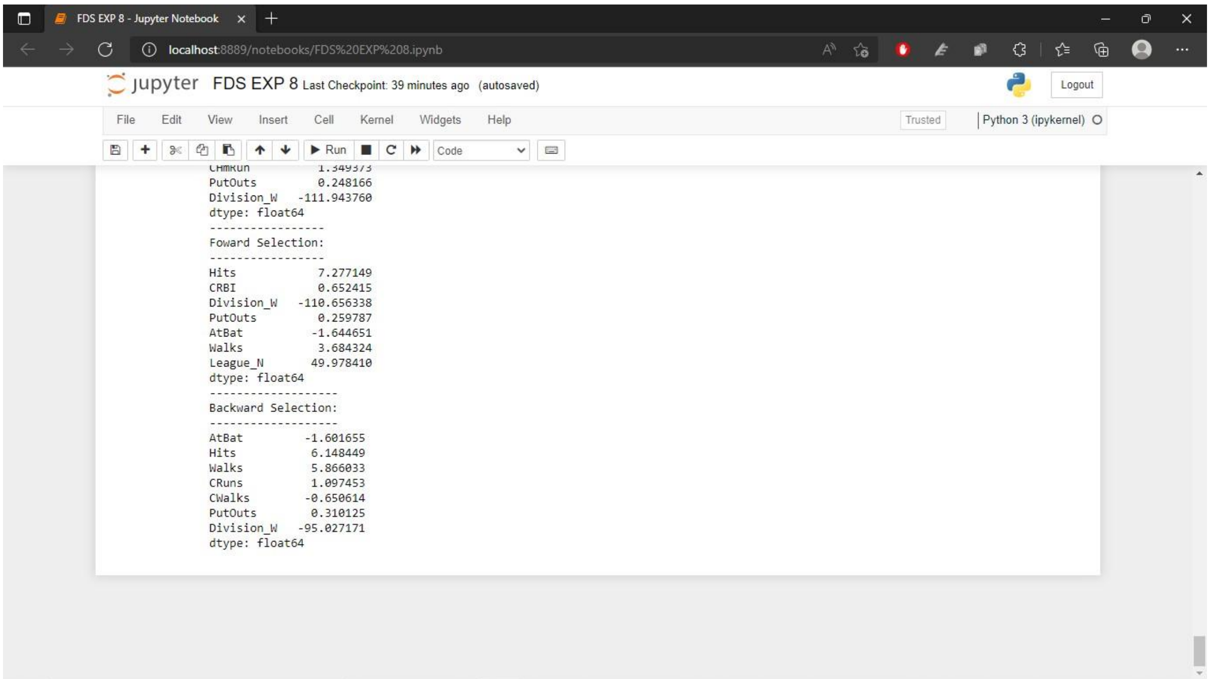


```
In [36]: models_bwd = pd.DataFrame(columns=["RSS", "model"], index = range(1,len(X.columns)))

tic = time.time()
predictors = X.columns

while(len(predictors) > 1):
    models_bwd.loc[len(predictors)-1] = backward(predictors)
    predictors = models_bwd.loc[len(predictors)-1]["model"].model.exog_names

toc = time.time()
print("Total elapsed time:", (toc-tic), "seconds.")
```



```
Processed  19 models on 18 predictors in 0.14936232566833496 seconds.
Processed  18 models on 17 predictors in 0.1008610725402832 seconds.
Processed  17 models on 16 predictors in 0.1591475009918213 seconds.
Processed  16 models on 15 predictors in 0.1291806697845459 seconds.
Processed  15 models on 14 predictors in 0.13464117050170898 seconds.
Processed  14 models on 13 predictors in 0.09548592567443848 seconds.
Processed  13 models on 12 predictors in 0.0921926498413086 seconds.
Processed  12 models on 11 predictors in 0.09603762626647949 seconds.
Processed  11 models on 10 predictors in 0.09082317352294922 seconds.
Processed  10 models on 9 predictors in 0.0801537036895752 seconds.
Processed  9 models on 8 predictors in 0.08101487159729004 seconds.
Processed  8 models on 7 predictors in 0.07615447044372559 seconds.
Processed  7 models on 6 predictors in 0.056688785552978516 seconds.
Processed  6 models on 5 predictors in 0.051862239837646484 seconds.
Processed  5 models on 4 predictors in 0.031914472579956055 seconds.
Processed  4 models on 3 predictors in 0.021941184997558594 seconds.
Processed  3 models on 2 predictors in 0.019945621490478516 seconds.
Processed  2 models on 1 predictors in 0.013962984085083008 seconds.
Total elapsed time: 1.5234317779541016 seconds.

In [37]: print("-----------")
```





Questions:

1. Explain other data reduction techniques in brief.

Data reduction aims to define it more compactly. When the data size is smaller, it is simpler to apply sophisticated and computationally high-priced algorithms. The reduction of the data may be in terms of the number of rows (records) or terms of the number of columns (dimensions).

There are various strategies for data reduction which are as follows −

Data cube aggregation − In this method, where aggregation operations are used to the data in the construction of a data cube. These data include the All Electronics sales per quarter, for the years 2002 to 2004. It is interested in the annual sales (total per year), rather than the total per quarter. Thus the data can be aggregated so that the resulting data summarize the total sales per year instead of per quarter. The resulting data set is smaller in volume, without loss of data essential for the analysis task.

Attribute subset selection − In this method, where irrelevant, weakly relevant, or redundant attributes or dimensions can be discovered and deleted. Data sets for analysis can include hundreds of attributes, some of which can be irrelevant to the mining task or redundant. For instance, if the task is to arrange customers as to whether or not they are likely to purchase a popular new CD at All Electronics when notified of a sale, attributes such as the customer's telephone number are likely to be irrelevant, unlike attributes such as age or music_taste.

Dimensionality reduction − Encoding mechanisms are used to reduce the data set size. In dimensionality reduction, data encoding or transformations are applied to obtain a reduced or "compressed" representation of the original data. If the original data can be reconstructed from the compressed data without any loss of information, the data reduction is called lossless.

Numerosity reduction − The data are restored or predicted by alternative, smaller data representations including parametric models (which are required to save only the model parameters rather than the actual data) or nonparametric methods including clustering, sampling, and the use of histograms.

Discretization and concept hierarchy generation − In this method, where raw data values for attributes are replaced by ranges or higher conceptual levels. Data discretization is a form of numerosity reduction that is very beneficial for the automatic production of concept hierarchies. Discretization and concept hierarchy generation are dynamic tools for data mining, in that they enable the mining of data at various levels of abstraction.

Outcomes:

CO 3 Apply the transformations required on data to make it suitable for mining

---

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)

Thus , I have implemented the forward , backward and best subset selection using python.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

1. Han, Kamber, "Data Mining Concepts and Techniques", Morgan Kaufmann 3rd Edition
2. Subset Selection is a Python adaptation of p. 244-247 of "Introduction to Statistical Learning with Applications in R" by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. Adapted by R. Jordan Crouser at Smith College for SDS293: Machine Learning (Spring 2016).
3. Dataset: https://www.kaggle.com/code/omeryasirkucuk/salary-prediction-models-on-hitters-dataset/data?select=Hitters.csv