Model Selection Demo:

In [2]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from itertools import combinations
```

In [3]:
```
# Simulate data
np.random.seed(0)
n = 1000
p = 5
X = np.random.randn(n, p)
beta = np.array([3, 2, 0, 0, 0])  # Only two variables are nonzero
y = X @ beta + np.random.randn(n) * 0.5

# Add a constant to X for intercept
X = sm.add_constant(X)
```

In [4]:
```
# Best subset selection
def best_subset_selection(X, y):
    n, p = X.shape
    models = []

    for k in range(1, p + 1):  # Iterate over subset sizes
        for combo in combinations(range(1, p), k):  # Generate combinations
            combo = (0,) + combo  # Include the intercept
            X_subset = X[:, combo]
            model = sm.OLS(y, X_subset).fit()
            models.append((model, combo))

    return models
```

In [5]:
```
# Calculate metrics
def calculate_metrics(model, X, y):
    n = len(y)
    k = model.df_model  # Number of predictors, excluding intercept



    # AIC
    aic = model.aic

    # BIC
    bic = model.bic


    # PRESS (Prediction Sum of Squares)
    hat_matrix = X @ np.linalg.inv(X.T @ X) @ X.T
    residuals = model.resid
    press = np.sum((residuals / (1 - np.diag(hat_matrix))) ** 2)

    # Adjusted R-squared
```

```
    r2 = model.rsquared
    adj_r2 = 1 - (1 - r2) * (n - 1) / (n - k)

    return aic, bic, press, adj_r2, int(k) #dont consider intercept as a pre
```

In [6]:
```
# Run best subset selection
models = best_subset_selection(X, y)
```

In [7]:
```
# Store results in pd DataFrame
results = []
for model, combo in models:
    aic, bic, press, adj_r2, num_predictors = calculate_metrics(model, X[:,
    results.append({
        'Predictors': combo,
        'n_Predictors': num_predictors,
        'AIC': aic,
        'BIC': bic,
        'PRESS': press,
        'Adjusted R^2': adj_r2
    })

# Convert results to pd DataFrame
results_df = pd.DataFrame(results)
results_df = results_df.sort_values(by='n_Predictors').reset_index(drop=True
```

In [8]:
```
# Display our results
pd.set_option('display.max_columns', None)  # Show all columns
results_df #0 represents intercept - all models include intercept
```

Out[8]:

	Predictors	n_Predictors	AIC	BIC	PRESS	Adjusted R^2
0	(0, 1)	1	4168.840872	4178.656382	3785.152258	0.707679
1	(0, 2)	1	5038.813069	5048.628579	9034.712100	0.302274
2	(0, 3)	1	5398.209642	5408.025152	12939.470547	0.000532
3	(0, 4)	1	5397.641492	5407.457002	12933.138971	0.001100
4	(0, 5)	1	5398.175148	5407.990659	12939.770161	0.000567
5	(0, 4, 5)	2	5399.000246	5413.723512	12950.664816	0.000740
6	(0, 3, 5)	2	5399.630015	5414.353281	12957.963976	0.000111
7	(0, 3, 4)	2	5399.106491	5413.829757	12951.514629	0.000634
8	(0, 2, 5)	2	5039.863992	5054.587258	9043.968854	0.302238
9	(0, 2, 4)	2	5039.363033	5054.086299	9040.554097	0.302587
10	(0, 2, 3)	2	5040.795516	5055.518782	9051.813650	0.301587
11	(0, 1, 5)	2	4170.835013	4185.558279	3792.691273	0.707387
12	(0, 1, 4)	2	4170.831107	4185.554373	3793.241268	0.707389
13	(0, 1, 3)	2	4167.760183	4182.483448	3780.874128	0.708286
14	(0, 1, 2)	2	1424.115862	1438.839128	243.221770	0.981232
15	(0, 3, 4, 5)	3	5400.451808	5420.082829	12968.914241	0.000286
16	(0, 2, 4, 5)	3	5040.303544	5059.934565	9048.707510	0.302627
17	(0, 2, 3, 5)	3	5041.849107	5061.480128	9061.154207	0.301548
18	(0, 2, 3, 4)	3	5041.345866	5060.976887	9057.644658	0.301900
19	(0, 1, 4, 5)	3	4172.824529	4192.455550	3800.766596	0.707097
20	(0, 1, 2, 3)	3	1425.005414	1444.636436	243.514714	0.981234
21	(0, 1, 3, 4)	3	4169.750085	4189.381106	3789.024898	0.707996
22	(0, 1, 2, 5)	3	1425.430861	1445.061882	243.507077	0.981226
23	(0, 1, 2, 4)	3	1426.061285	1445.692306	243.710573	0.981215
24	(0, 1, 3, 5)	3	4169.751048	4189.382069	3788.374057	0.707996
25	(0, 2, 3, 4, 5)	4	5042.289172	5066.827948	9065.882384	0.301937
26	(0, 1, 2, 3, 4)	4	1426.950318	1451.489094	244.003823	0.981217
27	(0, 1, 2, 3, 5)	4	1426.300991	1450.839767	243.798024	0.981229
28	(0, 1, 2, 4, 5)	4	1427.357141	1451.895918	243.996967	0.981209

	Predictors	n_Predictors	AIC	BIC	PRESS	Adjusted R^2
29	(0, 1, 3, 4, 5)	4	4171.740037	4196.278814	3796.512782	0.707706
30	(0, 1, 2, 3, 4, 5)	5	1428.226372	1457.672904	244.288229	0.981211

In [ ]:

In [ ]: