Multiple Linear Regression 2

MSc Statistics

Continuing example time spent on social media site

- Decide to remove variable degree as it is a dummy variable that does not show a strong relationship with our response variable daily minutes.
- Degree variable was not significant when included in the full model.
- Re-run the linear model to only include number of friends and hours spent working:

$$\text{daily\_minutes} = \beta_0 + \beta_1 \text{ num\_friends} + \beta_2 \text{ hours\_worked} + e$$

```
X = site_df[["num_friends","hours_worked"]]
X = sm.add_constant(X)
y = site_df['daily_minutes']
model_mlr_red = sm.OLS(y, X).fit()
```

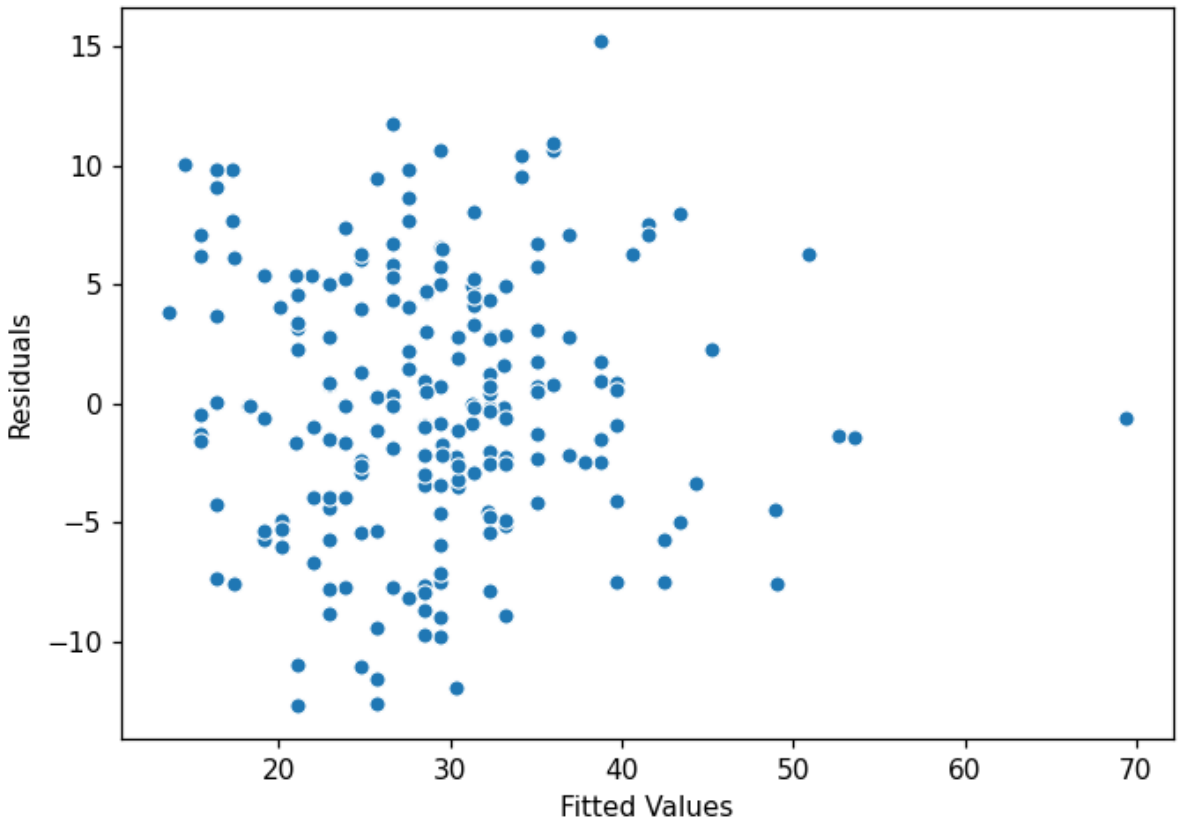
Continuing example time spent on social media site

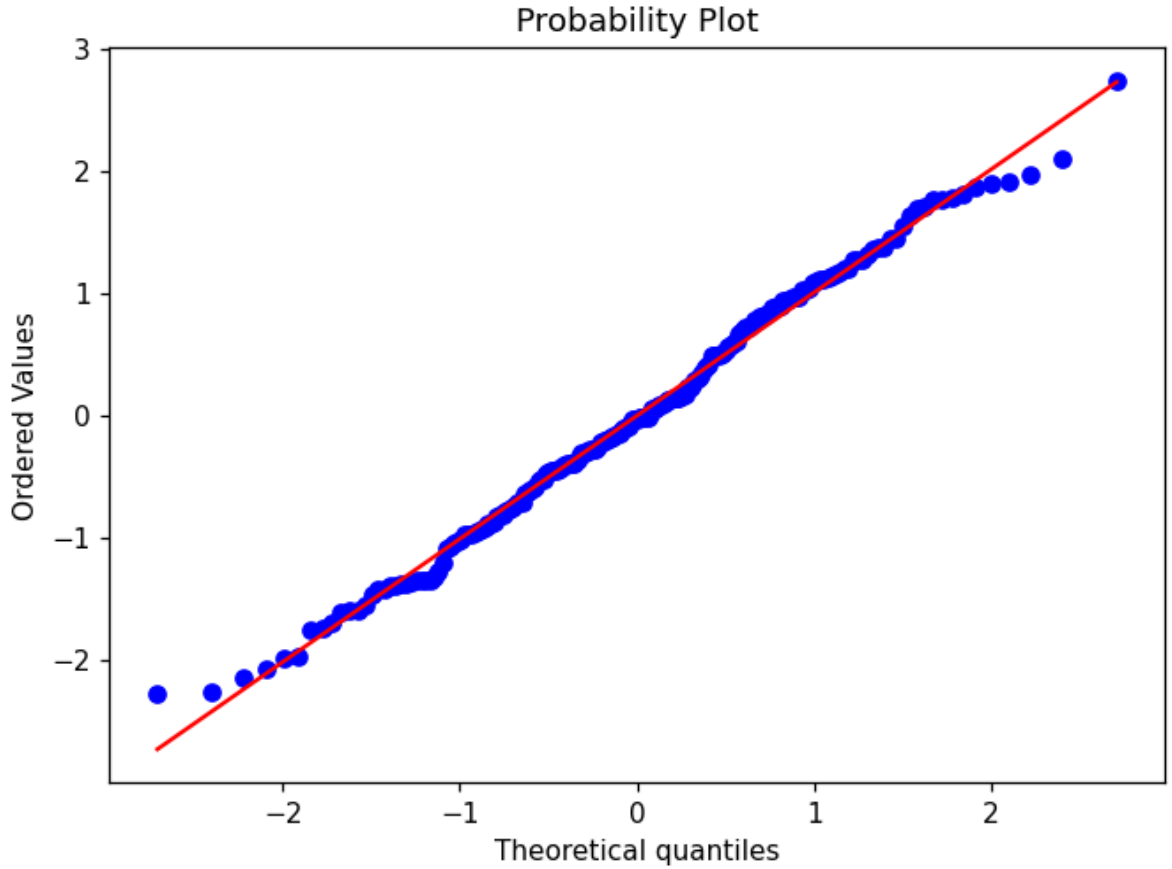
- Decide to remove variable degree as it is a dummy variable that does not show a strong relationship with our response variable daily minutes.
- Degree variable was not significant when included in the full model.
- Re-run the linear model to only include number of friends and hours spent working:

$$\text{daily\_minutes} = \beta_0 + \beta_1 \text{ num\_friends} + \beta_2 \text{ hours\_worked} + e$$

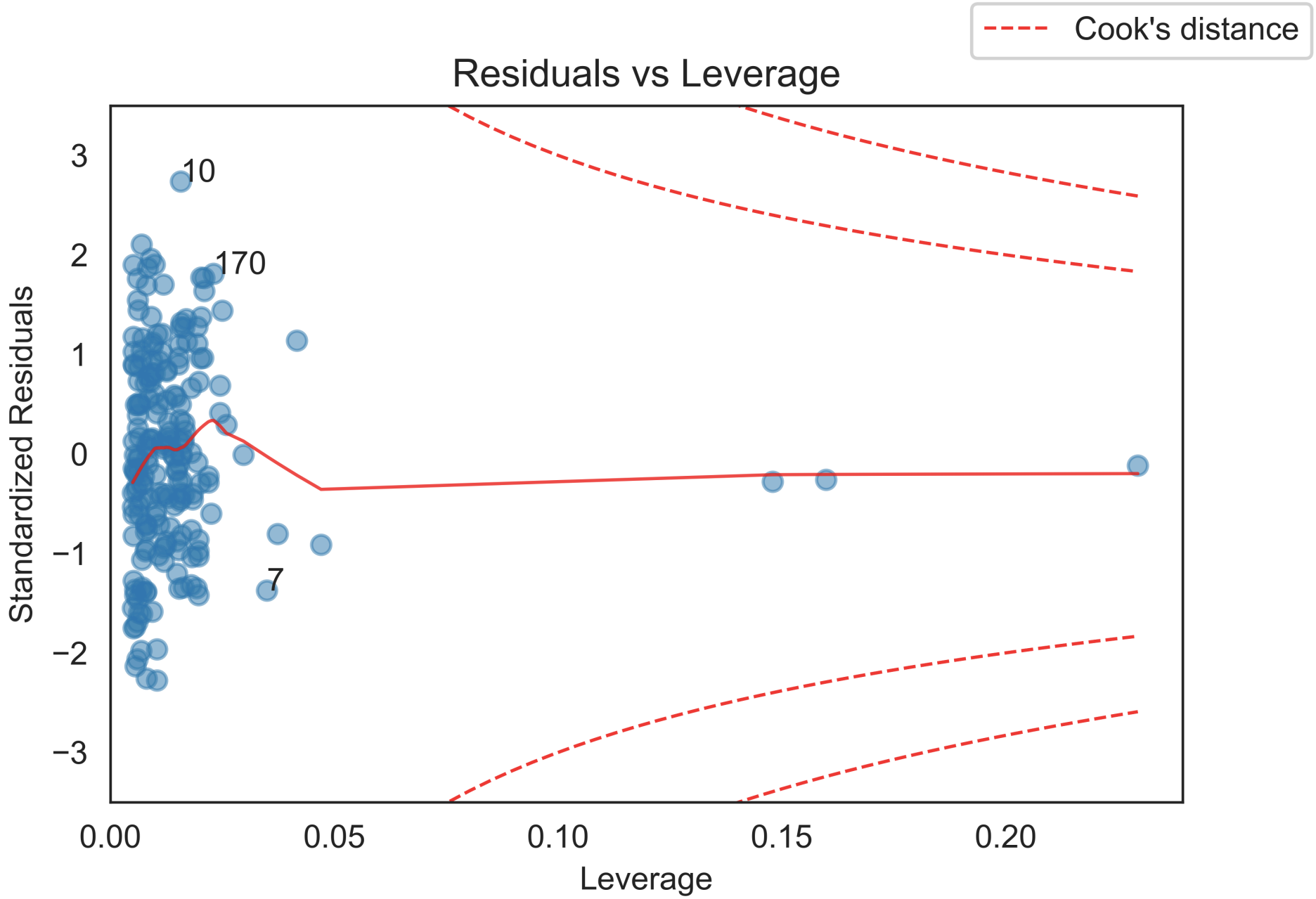
```
from statsmodels.formula.api import ols
model_mlr_red = ols('daily_minutes ~ num_friends+hours_worked', data=site_df).fit()
```

```
#Reminder of the code:

#fitted values
model_fitted_val = model_mlr_red.fittedvalues
#model residuals
model_residuals = model_mlr_red.resid
#standardised residuals
model_norm_residuals = model_mlr_red.get_influence().resid_studentized_internal

# leverage, from statsmodels internals
model_leverage = model_mlr_red.get_influence().hat_matrix_diag
# cook's distance, from statsmodels internals
model_cooks = model_mlr_red.get_influence().cooks_distance[0]

#create residual vs fitted values plot
sns.scatterplot(x=model_fitted_val,y=model_residuals)
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")

#create residual vs fitted values plot
stats.probplot(model_norm_residuals, plot=sns.mpl.pyplot)
plt.show()
```


Residuals
Fitted Values
15
10
5
0
−5
−10
20
30
40
50
60
70


Probability Plot
Ordered Values
Theoretical quantiles
3
2
1
0
−1
−2
−2
−1
0
1
2

```
plot_cooks = plt.figure();
plt.scatter(model_leverage, model_norm_residuals, alpha=0.5);
sns.regplot(model_leverage, model_norm_residuals,
            scatter=False, ci=False, lowess=True, line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8});
plot_cooks.axes[0].set_xlim(0, max(model_leverage)+0.01)
plot_cooks.axes[0].set_ylim(-3.5, 3.5)
plot_cooks.axes[0].set_title('Residuals vs Leverage')
plot_cooks.axes[0].set_xlabel('Leverage')
plot_cooks.axes[0].set_ylabel('Standardized Residuals');
# annotations
leverage_top_3 = np.flip(np.argsort(model_cooks), 0)[:3]
for i in leverage_top_3:
    plot_cooks.axes[0].annotate(i,
                                xy=(model_leverage[i],model_norm_residuals[i]));
p = len(model_mlr_red.params) # number of model parameters
graph(lambda x: np.sqrt((0.5 * p * (1 - x)) / x),
    np.linspace(0.001, max(model_leverage), 50), 'Cook\'s distance') # 0.5 line
graph(lambda x: np.sqrt((1 * p * (1 - x)) / x),np.linspace(0.001, max(model_leverage), 50)) # 1 line
graph(lambda x: -np.sqrt((0.5 * p * (1 - x)) / x),
    np.linspace(0.001, max(model_leverage), 50)) # 0.5 line
graph(lambda x: -np.sqrt((1 * p * (1 - x)) / x),
    np.linspace(0.001, max(model_leverage), 50)) # 1 line
plot_cooks.legend(loc='upper right');
plt.show()
```


Cook's distance
Residuals vs Leverage
Standardized Residuals
Leverage
3
2
1
0
−1
−2
−3
0.00
0.05
0.10
0.15
0.20
10
170
7

OLS Regression Results

Dep. Variable:	daily_minutes	R-squared:	0.679
Model:	OLS	Adj. R-squared:	0.675
Method:	Least Squares	F-statistic:	211.2
Date:	Wed, 1 Jan 2020	Prob (F-statistic):	5.01e-50
Time:	00:00:00	Log-Likelihood:	-637.05
No. Observations:	203	AIC:	1280.
Df Residuals:	200	BIC:	1290.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	31.3777	0.819	38.304	0.000	29.762	32.993
num_friends	0.9274	0.063	14.680	0.000	0.803	1.052
hours_worked	-1.8675	0.127	-14.749	0.000	-2.117	-1.618

Omnibus:	2.934	Durbin-Watson:	2.038
Prob(Omnibus):	0.231	Jarque-Bera (JB):	2.076
Skew:	0.030	Prob(JB):	0.354
Kurtosis:	2.508	Cond. No.	21.1

Notes:

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

Interpreting the Model:

- One additional friend corresponds to on average an extra minute spent on the site each day when number of hours worked is kept constant.

- Each additional hour in a user's workday corresponds to on average around two fewer minutes spent on the site each day when keeping the number of friends constant.

Fit of the model:

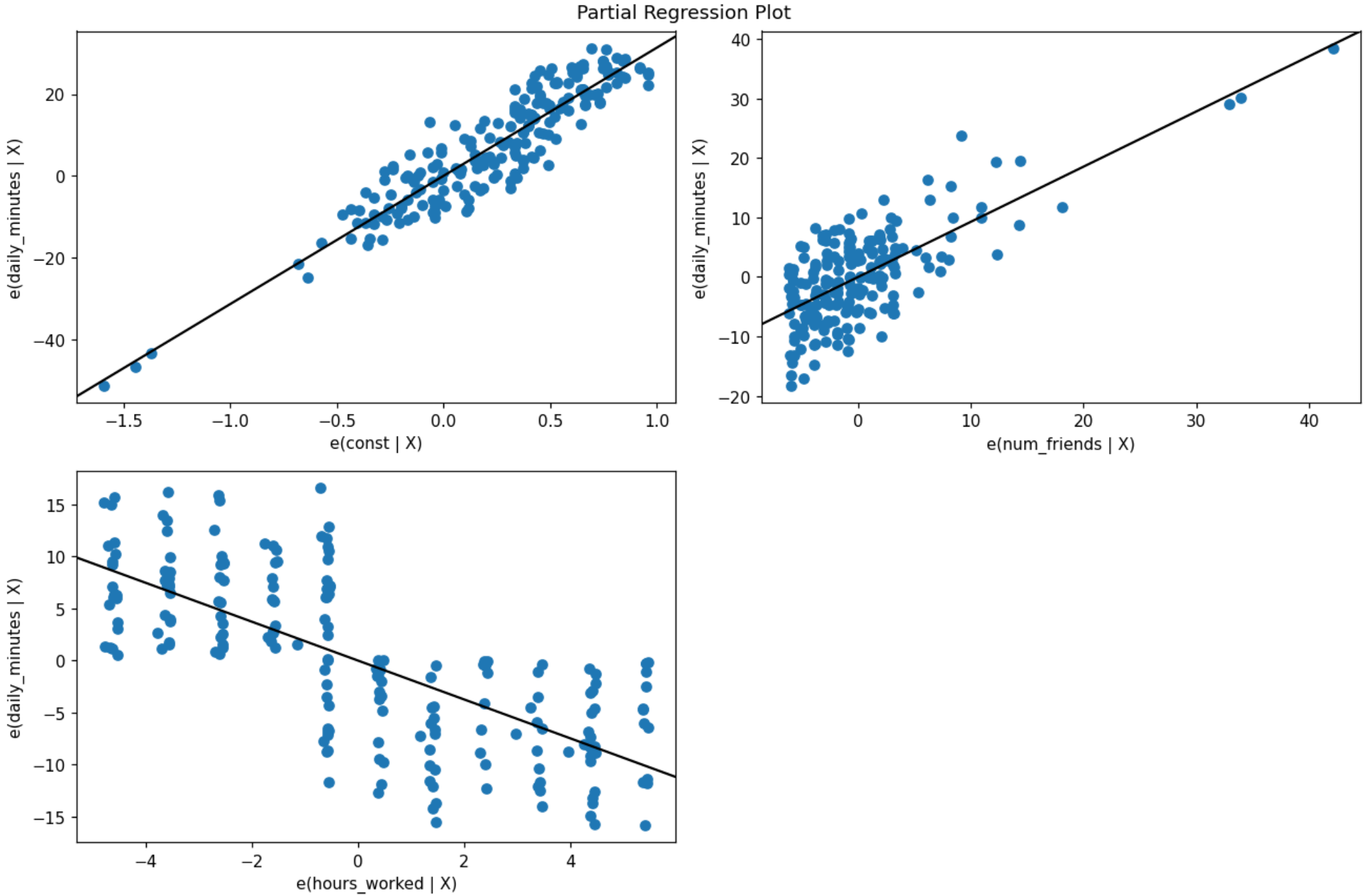
H_0: $\beta_i = 0$, β_i = i^{th} partial regression coefficient (partial slope)

H_1: $\beta_i \neq 0$

- Looking at the table of coefficients results:
  - $\hat{\beta}_1$, estimate for num_friends, has t-test value = 14.68, p-value <0.001.
    Therefore, reject the null hypothesis

  - $\hat{\beta}_2$, estimate for hours_worked, has t-test value = -14.75, p-value <0.001.
    Therefore, reject the null hypothesis

```
def get_partial_regression_plot(fitted_model, figure_size=(12, 8), save_to_file=False,
file_name="regression_plot"):

    reg_plot = plot_partregress_grid(fitted_model, fig=plt.figure(figsize=figure_size))

    if save_to_file:
        reg_plot.savefig(file_name + ".png")

    return reg_plot

reg_plot = get_partial_regression_plot(model_mlr_red, save_to_file=True)
plt.show()
```


Partial Regression Plot
e(daily_minutes | X)
e(const | X)
e(num_friends | X)
e(hours_worked | X)

Fit of the model:

- R-squared value is 67.9%
- **Note: The R-squared can never increase when a variable is removed.**
- The R-squared value decreased from .1% by removing Degree variable, indicating very little of the variation in time spent on site was being explained by Degree.
- The adjusted R-squared is a modified version of R-squared that takes into account the number of predictors in the model.

R-squared vs Adjusted R-squared

- Every time you add a predictor to a model, the R-squared increases, even if due to chance alone. It never decreases. Consequently, a model with more terms may appear to have a better fit simply because it has more terms.
- If a model has too many predictors and higher order polynomials, it begins to model the random noise in the data. This condition is known as **overfitting the model** and it produces misleadingly high R-squared values and a lessened ability to make predictions.
- The adjusted R-squared compensates for the addition of variables and only increases if the new predictor enhances the model above what would be obtained by chance. Conversely, it will decrease when a predictor improves the model less than what is predicted by chance.

R-squared Vs Adjusted R-squared

Model	**R-Squared**	**Adjusted R-Squared**
daily_minutes = $\beta_0 + \beta_1$ num_friends + e	0.329	0.326
daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked + e	0.679	0.675
daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked + β_3 degree + e	0.68	0.675
	R-squared always increases as more predictors added	Adjusted R-squared does not always increases as more predictors added, can be used to see which model performs better.

```
model_mlr_red.scale
31.606703176644857

np.sqrt(model_mlr_red.scale)
5.621983918212934
```

- The standard error of the regression provides the absolute measure of the typical distance that the data points fall from the regression line. S is in the units of the dependent variable.

Model	**Residual Standard Error**
daily_minutes = $\beta_0 + \beta_1$ num_friends + e	8.103
daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked + e	5.622
daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked + β_3 degree + e	5.624

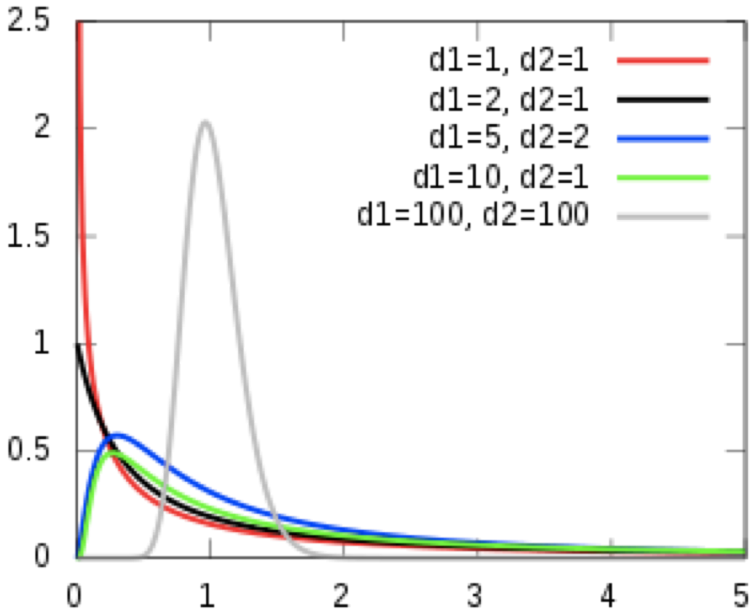
- The residual standard error is lower again suggesting better fit

F-test for fit of model

- F-test used to compares two models
- Output by python, F-test compares the model performed to the null model, the model that only includes the intercept.
- i.e. test if the independent variables describe the dependent variable well.
- The null and alternative hypothesis are as follows:
- H_0: $\beta_1 = \beta_2 = \ldots = \beta_k = 0$, for k variables in the model
- H_1: Not all β_i are 0.

F Distribution

- A continuous statistical distribution which arises in the testing of whether two observed samples have the same variance.
- Parameter that describes this distribution is degrees of freedom (n-p) from both samples.



F-test for fit of model

OLS Regression Results

Dep. Variable:	daily_minutes	R-squared:	0.679
Model:	OLS	Adj. R-squared:	0.675
Method:	Least Squares	F-statistic:	211.2
Date:	Wed, 1 Jan 2020	Prob (F-statistic):	5.01e-50
Time:	00:00:00	Log-Likelihood:	-637.05
No. Observations:	203	AIC:	1280.
Df Residuals:	200	BIC:	1290.
Df Model:	2		
Covariance Type:	nonrobust		

- H_0: $\beta_1 = \beta_2 = 0$, β_1 coefficient for num_friends, β_2 coefficient for hours_worked
- H_1: Not all β_i are 0, where i = [1,2]
- F-statistic: 211.2, p-value: < 5.01e-50
- P-value < 0.05, therefore, reject H_0 and conclude not all the coefficients in the model are 0

Testing two models in python

- Full model: daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked+ β_3 degree
- Reduced model: daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked

- H_0: $\beta_3 = 0$
- H_1: $\beta_3 \neq 0$

Testing two models in python

```
from statsmodels.stats.anova import anova_lm

#run anova test on multiple linear regressions with the smaller model first
anovaResults = anova_lm(model_mlr_red, model_mlr)
print(anovaResults)
   df_resid         ssr  df_diff    ss_diff        F   Pr(>F)
0     200.0  6321.340635      0.0        NaN      NaN      NaN
1     199.0  6294.264579      1.0  27.076057  0.856039  0.35597
```

Testing two models in python

- F-statistic = 0.856 and p-value = 0.356
- p-value > 0.05, therefore, fail to reject H_0 that β_3 coefficient for degree is 0.
- This is the same p-value from the t-test in the coefficient table for the full model.

- Use F-test to compare more than one parameter at a time.

Model Selection

- Suppose there are k available predictors. Then there are 2^k possible model choices (discounting transformations, interactions etc)
- When a predictor is added:

1. SSE (sum of squares of errors) decreases or stays the same.
2. $R^2 = 1 - \frac{SSE}{SST}$ increases, or stays the same. (SST = Total Sum of Squares)
3. MSE= $\frac{SSE}{n-p}$ usually decreases. (as p increases by 1, n-p decreases).

   MSE (Mean Square Error) will decrease unless the decrease in SSE is not enough to compensate for the loss of one degree of freedom.

Comparing Models

- Generally, we would like models with low SSE and low p (number of parameters.) Here are some measures used to compare models.

1. **AIC** - Akaike Information Criterion

$$AIC = n \log\left(\frac{SSE}{n}\right) + p$$

Models with small values of AIC are preferred.

2. **BIC** - Bayes Information Criterion

$$BIC = n \log\left(\frac{SSE}{n}\right) + p \log(n)$$

Models with small values of BIC are preferred.

AIC and BIC in python

OLS Regression Results

Dep. Variable:	daily_minutes	R-squared:	0.679
Model:	OLS	Adj. R-squared:	0.675
Method:	Least Squares	F-statistic:	211.2
Date:	Wed, 1 Jan 2020	Prob (F-statistic):	5.01e-50
Time:	00:00:00	Log-Likelihood:	-637.05
No. Observations:	203	AIC:	1280.
Df Residuals:	200	BIC:	1290.
Df Model:	2		
Covariance Type:	nonrobust		

Model	**AIC**	**BIC**
daily_minutes = $\beta_0 + \beta_1$ num_friends + e	1428	1434
daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked + e	1280	1290
daily_minutes = $\beta_0 + \beta_1$ num_friends + β_2 hours_worked + β_3 degree + e	1281	1294

Backward Elimination/Selection:

1. Select significance level
2. Fit our model with all possible independent variables.
3. Consider variable with highest p-value.
4. If p-value is greater than significance level, remove variable
5. Again fit the model without removed variable.
6. Repeat steps 2-5 until happy with the fit of model.

Forward Elimination/Selection:

1. Select significance level
2. Fit intercept-only model.
3. Find the independent variable that is not included in the model that when added has smallest p-value and add it if p-value is less than significance level.
4. Re-fit the model and repeat step 3 until all independent variables that have not been added have a p-value above the threshold.

Important points:

- Backward/Forward Elimination is not ideal solution and need to take in account other considerations like for instance:
- If any independent variables are correlated with each other, remove one of these
- Increasing the number of tests performed inflates the type I error rate. I.e. increases the chance of finding something ‘significant’ just by chance.
- If there are many variables, looking at all possible models may not be the best thing to do.