ORIE 3120: Practical Tools for OR, DS, and ML

Linear Regression Basics (Lecture 11)

Professor Frazier

2022

Announcements

- Project Teams: Google form posted later today, due March 11.
- Other project deadlines:
  - Milestone 1 due 4/18
  - Peer grading for milestone 1 due shortly afterward (date TBD)
  - Final Project due 5/16
  - Peer grading for final project due 5/21
- In nearly all cases, points earned on project team output (milestone 1, final report) will be the same for all students on the team
- BUT it sometimes happens that one person on a team does little or nothing
- Collaboration tools that track who did what are strongly encouraged: Google docs, google collab, and github all do this
- Google form will ask for links to a Google Drive folder and/or GitHub repository that let the teaching team check contributions by individual team members

Outline

Introduction to Linear Models

- Why use regression analysis?
- Linear models

Estimation

- Data
- Least squares

Example: Electricity Usage I

- Description
- Lecture questions
- Leaving off the intercept
- Reading model output
- Statistical output
- Multiple R Squared
- Adding temperature squared

Part 1: Linear Regression - Introduction

Regression can

- investigate how variables are related,
- predict future values of variables, and
- show how output variables will change if we change input variables.

. . . the last step can be useful for control!

Regression examples (COVID-19 at Cornell)

- Of 86 student COVID-19 cases in the fall 2020 semester, 10% were found immediately after travel
- BUT, members of more social student cohorts traveled more *and* socialized more locally
- Does travel cause COVID? Or do people with infection-prone local behavior also travel?
- The answer: After controlling for other risk factors with regression, we found that traveling in the fall 2020 semester elevates a students' odds of testing positive by 8x.

Link: Mathematical Modeling for Cornell's 2021 Spring Semester

Regression examples (from Kaggle)

kaggle™

Sign Up | About Kaggle | Create a competition | Competitions | Forums | Blog | Careers

We're making data science a sport.™

Participate in competitions

Kaggle is an arena where you can match your data science skills against a global cadre of experts in statistics, mathematics, and machine learning. Whether you're a world-class algorithm wizard competing for prize money or a novice looking to learn from the best, here's your chance to jump in and geek out, for fame, fortune, or fun.

Sign up as a competitor

(Need convincing?)

Create a competition

Kaggle is a platform for data prediction competitions that allows organizations to post their data and have it scrutinized by the world's best data scientists. In exchange for a prize, winning competitors provide the algorithms that beat all other methods of solving a data crunching problem. Most data problems can be framed as a competition.

See how it works

Featured Competitions

Browse all »

Heritage Health Prize
Identify patients who will be admitted to a hospital within the next year, using historical claims data.
Ends **12 months** · **905** teams · **$3 million**

ASAP
Automated Student Assessment Prize
Phase One: Automated Essay Scoring

The Hewlett Foundation: Automated Essay Scoring
Develop an automated scoring algorithm for student-written essays.
Ends **33 days** · **94** teams · **$100,000**

Boehringer Ingelheim

Predicting a Biological Response
Predict a biological response of molecules from their chemical properties
Ends **2 months** · **122** teams · **$20,000**

Username

Sign in | Remember me?

Forgot your username/password?

View available positions »

Regression for insurance claim predictions

Claim Prediction Challenge (Allstate)

Prize pool	Teams	Completed
$10,000	107	5 months ago

Many factors contribute to the frequency and severity of car accidents including how, where and under what conditions people drive, as well as what they are driving.

Bodily Injury Liability Insurance covers other people's bodily injury or death for which the insured is responsible. **The goal of this competition is to predict Bodily Injury Liability Insurance claim payments based on the characteristics of the insured's vehicle.**

Regression for finance

Benchmark Bond Trade Price Challenge

Prize pool	Teams	Ends
$17,500	168	33 days

The Benchmark Bond Trade Price Challenge is a competition to predict the next price that a US corporate bond might trade at. Contestants are given information on the bond including current coupon, time to maturity and a reference price computed by **Benchmark Solutions**. Details of the previous 10 trades are also provided.

Benchmark Solutions

Regression for social network analysis

Wikipedia's Participation Challenge

Prize pool	Teams	Completed
$10,000	96	6 months ago

Wikipedia is the encyclopedia that can be edited by everybody but a majority of our editors make few edits (less than 50). We want to grow our editor community in terms of diversity and size so we need to understand why editors stop editing.

Between 2005 and 2007, newbies started having real trouble successfully joining the Wikimedia community. Before 2005 in the English Wikipedia, nearly 40% of new editors would still be active a year after their first edit. After 2007, only about 12-15% of new editors were still active a year after their first edit. Post-2007, lots of people were still trying to become Wikipedia editors. What had changed, though, is that they were increasingly failing to integrate into the Wikipedia community, and failing increasingly quickly. The Wikimedia community had become too hard to penetrate. For more information about these trends, please have a look at http://strategy.wikimedia.org/wiki/March_2011_Update

We want a quantitative answer to what factors predict future editing behavior, in order to ensure that the content of Wikipedia continues to grow both quantitively and qualitatively.

Regression setup

we want to predict output given inputs

- p input variables $X_1, \ldots, X_p \in \mathbf{R}$
  - also called "predictors", "independent variables", "covariates"
- output variable $Y \in \mathbf{R}$
  - also called "outcome", "response", "dependent variable", "label", "target" ...

In the Allstate example:

- Y is the cost of an insurance claim.
- $X_1, \ldots, X_p$ are the properties of the insured and his/her vehicle, e.g., credit score, age of the vehicle, ...

Linear models

linear model assumption: for some **parameters** $\beta_0, \ldots, \beta_p \in \mathbf{R}$,

$$E(Y|X_1, \ldots, X_p) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

hence

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

where ϵ is variation of Y about $E(Y|X_1, \ldots, X_p)$

- β_0 is the **intercept**
- $\beta_1, \ldots, \beta_p$ are called **regression coefficients**
  - also called "slopes" or "partial derivatives":

$$\beta_j = \frac{\partial}{\partial X_j} E(Y|X_1, \ldots, X_p)$$

- ϵ is the unpredictable variation in Y
  - also called the "noise", "error", or "residual variation"

Outline

Introduction to Linear Models

- Why use regression analysis?
- Linear models

Estimation

- Data
- Least squares

Example: Electricity Usage I

- Description
- Lecture questions
- Leaving off the intercept
- Reading model output
- Statistical output
- Multiple R Squared
- Adding temperature squared

Data

observe $(Y_i, X_{i,1}, \ldots, X_{i,p})$, for $i = 1, \ldots, n$

- n observations total
- i = index of "observation" = "subject" = "row in data spreadsheet"
- so the linear regression model can be rewritten as

$$Y_i = \beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + \cdots + \beta_p X_{i,p} + \epsilon_i$$

- notice that $\beta_0, \beta_1, \ldots, \beta_p$ do not depend on i
- the columns of the data spreadsheet are $Y_i, X_{i,1}, \ldots, X_{i,p}$

i	Y_i	$X_{i,1}$	$X_{i,2}$	$\ldots$	$X_{i,p}$
1	2.3	1.1	6.2	$\ldots$	5.9
2	12.7	2.4	5.4	$\ldots$	9.6
3	6.3	0.9	6.9	$\ldots$	1.5

Least squares estimation

What if we don't know the output Y_i for some subject i? Predict it!

- Given estimates $\hat{\beta}_0, \ldots, \hat{\beta}_p$ for the regression coefficients, predict Y_i as
$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_{i,1} + \hat{\beta}_2 X_{i,2} + \cdots + \hat{\beta}_p X_{i,p}$$
- $\hat{Y}_i$ is an estimate of
$E(Y_i | X_{i,1}, \ldots, X_{i,p}) = \beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + \cdots + \beta_p X_{i,p}$
- $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ find the "best" predictor by minimizing
$$\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

Residuals and Fitted Values

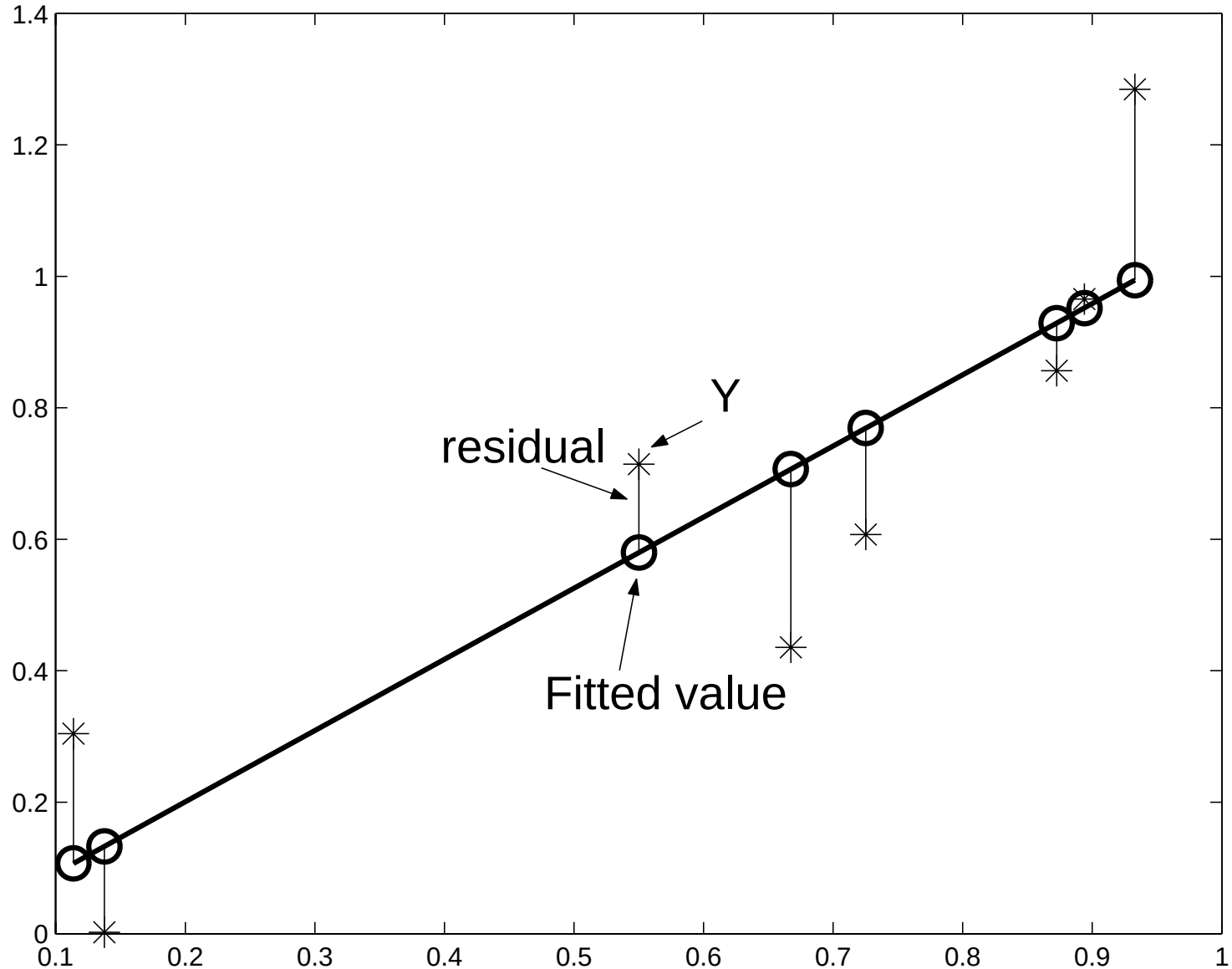
$$\begin{aligned} \text{Fitted value:} \quad \hat{Y}_i &= \hat{\beta}_0 + \hat{\beta}_1 X_{i,1} + \hat{\beta}_2 X_{i,2} + \cdots + \hat{\beta}_p X_{i,p} \\ \text{Residual:} \quad \widehat{\epsilon}_i &= Y_i - \hat{Y}_i \qquad (\text{estimates } \epsilon_i) \end{aligned}$$

Least-squares: makes the sum of square residuals

$$\sum_{i=1}^{n} \widehat{\epsilon}_i^2$$

as small as possible.

Least Squares, in pictures



When is a model linear?

Linear regression assumes

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon.$$

Definition: A model is **linear** if Y is linear in the **parameters** (linear in $\beta_0, \ldots, \beta_p$).

Key point: Covariates $X_1, \ldots, X_p$ can be anything observable. They can be nonlinear functions of measured quantities.

Nonlinear models

- Nonlinear regression (including deep learning) is covered in more advanced courses
- For this course, you only need to know a nonlinear model when you see it

Example:

$$Y = \beta_0 + \beta_1 \exp(\beta_2 X) + \epsilon$$

Nonlinear models

- Nonlinear regression (including deep learning) is covered in more advanced courses
- For this course, you only need to know a nonlinear model when you see it

Example:

$$Y = \beta_0 + \beta_1 \exp(\beta_2 X) + \epsilon$$

This model is nonlinear because Y is a nonlinear function of β_2.

Nonlinear models

- Nonlinear regression (including deep learning) is covered in more advanced courses
- For this course, you only need to know a nonlinear model when you see it

Example:

$$Y = \beta_0 + \beta_1 \exp(\beta_2 X) + \epsilon$$

This model is nonlinear because Y is a nonlinear function of β_2.

Example:

$$Y = \beta_0 + \beta_1^3 \beta_2^4 \exp(X) + \epsilon$$

Nonlinear models

- Nonlinear regression (including deep learning) is covered in more advanced courses
- For this course, you only need to know a nonlinear model when you see it

Example:

$$Y = \beta_0 + \beta_1 \exp(\beta_2 X) + \epsilon$$

This model is nonlinear because Y is a nonlinear function of β_2.

Example:

$$Y = \beta_0 + \beta_1^3 \beta_2^4 \exp(X) + \epsilon$$

This model is nonlinear in the parameters, but can be rewritten as a linear model in $\beta_1' = \beta_1^3 \beta_2^4$

$$Y = \beta_0 + \beta_1' \exp(X) + \epsilon$$

Q1: Which of these models are linear in the parameters?

$$1. Y = \beta_0 + \beta_1 \times (\text{CreditScore}) + \beta_2 \times (\text{CreditScore})^2,$$

$$2. Y = \beta_0 + \beta_1 \times (\text{CreditScore}) + (\beta_2 \times \text{CreditScore})^2,$$

$$3. Y = \beta_0 + \beta_1 \times (\text{CreditScore}) + \exp(\beta_2 \times \text{CreditScore}),$$

- A. none of them
- B. 1 only
- C. 2 only
- D. 3 only
- E. more than one

Q2: Which of these can be rewritten as a linear model?

$$1.\, Y = \beta_0 + \beta_1 \times (\text{CreditScore}) + \beta_2 \times (\text{CreditScore})^2,$$

$$2.\, Y = \beta_0 + \beta_1 \times (\text{CreditScore}) + (\beta_2 \times \text{CreditScore})^2,$$

$$3.\, Y = \beta_0 + \beta_1 \times (\text{CreditScore}) + \exp(\beta_2 \times \text{CreditScore}),$$

- A. none of them
- B. 1 only
- C. 2 only
- D. 3 only
- E. more than one

A guide to python packages we'll use in this unit

- sqlite: interface with an SQLite database
- numpy (np): mathematical operations (esp. linear algebra)
- scipy: more mathematical operations (including statistics)
- pandas (pd): manipulate data tables (dataframes)
- matplotlib (plt): for plotting
- seaborn: for statistical plots
- sklearn: for machine learning (regression and beyond)
- statsmodels (sm): statistical models

Outline

Introduction to Linear Models

- Why use regression analysis?
- Linear models

Estimation

- Data
- Least squares

Example: Electricity Usage I

- Description
- Lecture questions
- Leaving off the intercept
- Reading model output
- Statistical output
- Multiple R Squared
- Adding temperature squared

Example: Electricity Usage

- We are managing a large complex of apartments in the Northeast.
- We pay for the electricity used by our residents.
- We would like to predict electricity usage so that we can estimate how much money should be set aside.

Demo on the course webpage

Predictors (Independent Variables)

The following predictors were measured during data collection:

- year (1989 – 1994)
- month (1 – 12)
- usage (of electricity for month)
- temperature (average temperature for month)

We added two additional predictors:

- yearcts = year + (month - 1)/12.
- tempsqr = temperature^2.

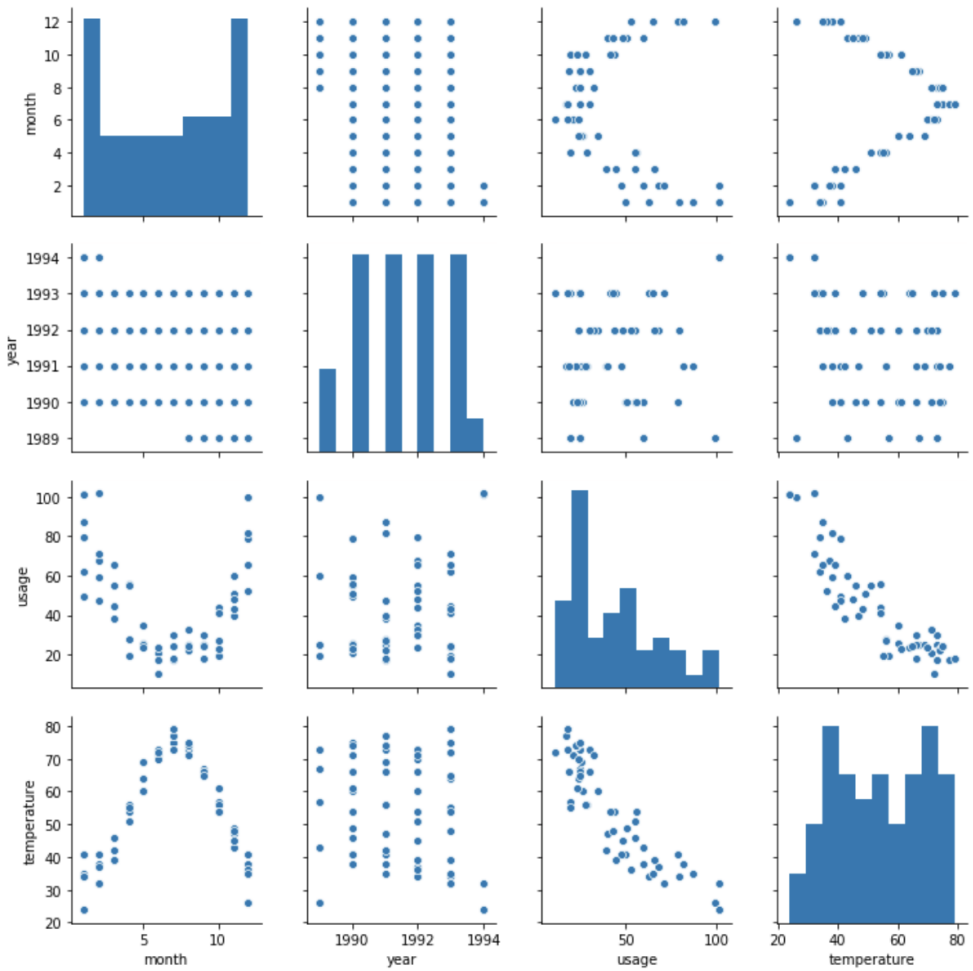
Let's look at the demo

In part 1 of the demo, we'll:

- visualize the data
- get a little more experience manipulating dataframes
- see how to run regressions in python
- get some experience seeing how regressions change based on the predictors used
- see how to use formulas and statsmodels.formula.api to run regressions more easily

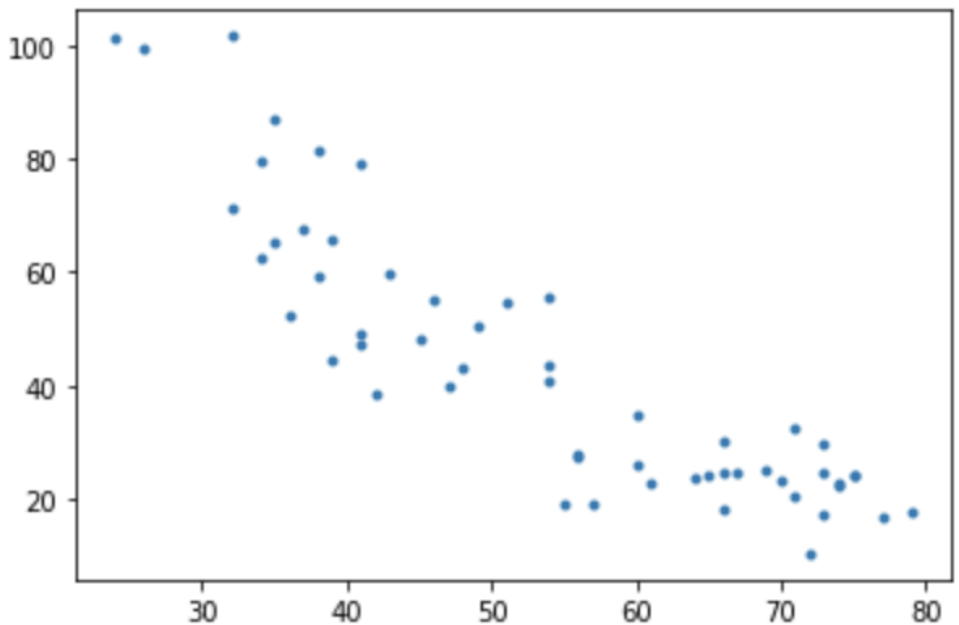
Load data, Plot it

```
usage = pd.read_csv('elec_usage.txt')
seaborn.pairplot(usage)
```



Plot data

```
plt.plot(usage['temperature'],usage['usage'],'.')
```

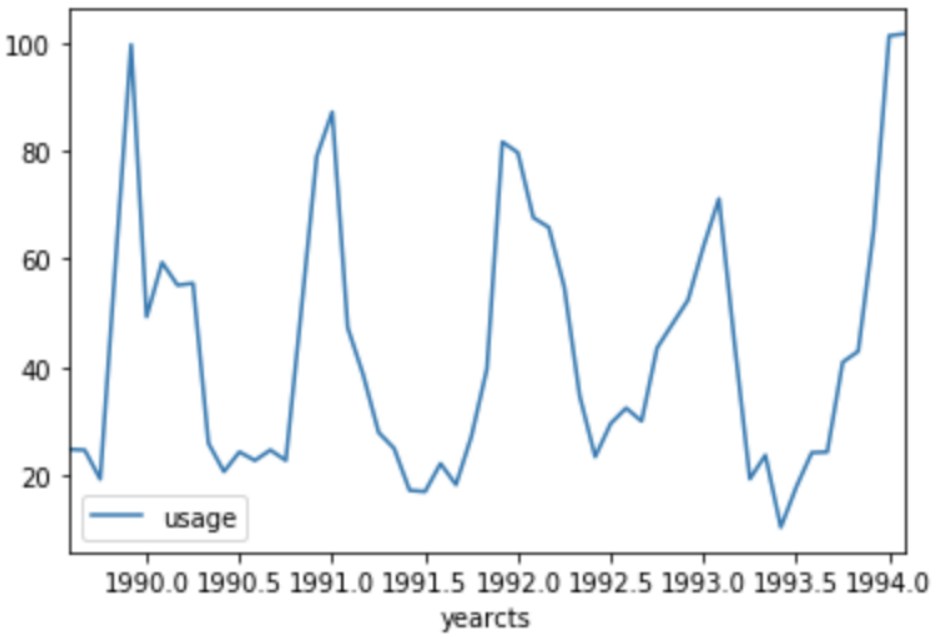


Create extra features

```
usage['tempsqr'] = usage['temperature']^2
usage['yearcts'] = usage['year'] + (usage['month']-1)/12
```

Plot data

```
usage.plot.line(x='yearcts',y='usage')
```



Preparing to do regression with statsmodels

Recall that linear regression assumes:

$$Y_i = \beta_0 + \beta_1 X_{i,1} + \cdots + \beta_p X_{i,p} + \epsilon_i$$

for each datapoint $i = 1, 2, \ldots, n$.

We can rewrite this as

$$Y = X\beta + \epsilon$$

where:

- $Y = [Y_1, \ldots, Y_n]^T$ is an $n \times 1$ column vector
- $\beta = [\beta_0, \beta_1, \ldots, \beta_p]^T$ is an $(p+1) \times 1$ column vector
- X is an $n \times (p+1)$ matrix whose rows are $[1, X_{i,1}, ..., X_{i,p}]$.
- $\epsilon = [\epsilon_1, \ldots, \epsilon_n]^T$ is an $n \times 1$ column vector

Preparing to do regression with statsmodels

We can write the assumption of linear regression as

$$Y = X\beta + \epsilon$$

$$\begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix} = \begin{pmatrix} 1 & X_{1,1} & \cdots & X_{1,p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{n,1} & \cdots & X_{n,p} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{pmatrix}$$

Preparing to do regression with statsmodels

We create a Series `Y` and a Dataframe `X` to pass as input to `statsmodels`

```
Y = usage['usage']
```

```
Y.head()
```

```
0    24.828
1    24.688
2    19.310
3    59.706
4    99.667
Name: usage, dtype: float64
```

Preparing to do regression with statsmodels

We create a Series `Y` and a Dataframe `X` to pass as input to `statsmodels`

```
X = usage['temperature']
X = sm.add_constant(X)
```

```
X.head()
```

	const	temperature
0	1.0	73
1	1.0	67
2	1.0	57
3	1.0	43
4	1.0	26

Fitting a linear regression model with statsmodels

Now that we have the Series `Y` and the Dataframe `X`, we can fit linear regression

```
model = sm.OLS(Y, X).fit()
```

Lecture Question 3

```
n = 1000
# np.random.randn(n) generates a series
# of n Normal(0,1) random variables
X = np.random.randn(n)
X = sm.add_constant(X)
eps = np.random.randn(n)
Y = 1 + eps
model = sm.OLS(Y, X).fit()
```

What are the correct values for β_0 and β_1?

- A. $\beta_0 = 0$, $\beta_1 = 0$
- B. $\beta_0 = 0$, $\beta_1 = 1$
- C. $\beta_0 = 1$, $\beta_1 = 0$
- D. $\beta_0 = 1$, $\beta_1 = 1$
- E. None of the above

Getting predictions from the fitted model

```
Yhat = model.predict(X)
```

```
Yhat.head()
```

```
0    17.099345
1    25.287030
2    38.933173
3    58.037772
4    81.236215
dtype: float64
```

Getting statistics from the fitted model

```
# Print out the statistics
model.summary()
```

OLS Regression Results

Dep. Variable:	usage	**R-squared:**	0.780
Model:	OLS	**Adj. R-squared:**	0.776
Method:	Least Squares	**F-statistic:**	188.4
Date:	Wed, 09 Mar 2022	**Prob (F-statistic):**	4.35e-19
Time:	08:26:27	**Log-Likelihood:**	-210.65
No. Observations:	55	**AIC:**	425.3
Df Residuals:	53	**BIC:**	429.3
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	116.7162	5.565	20.973	0.000	105.554	127.878
temperature	-1.3646	0.099	-13.727	0.000	-1.564	-1.165

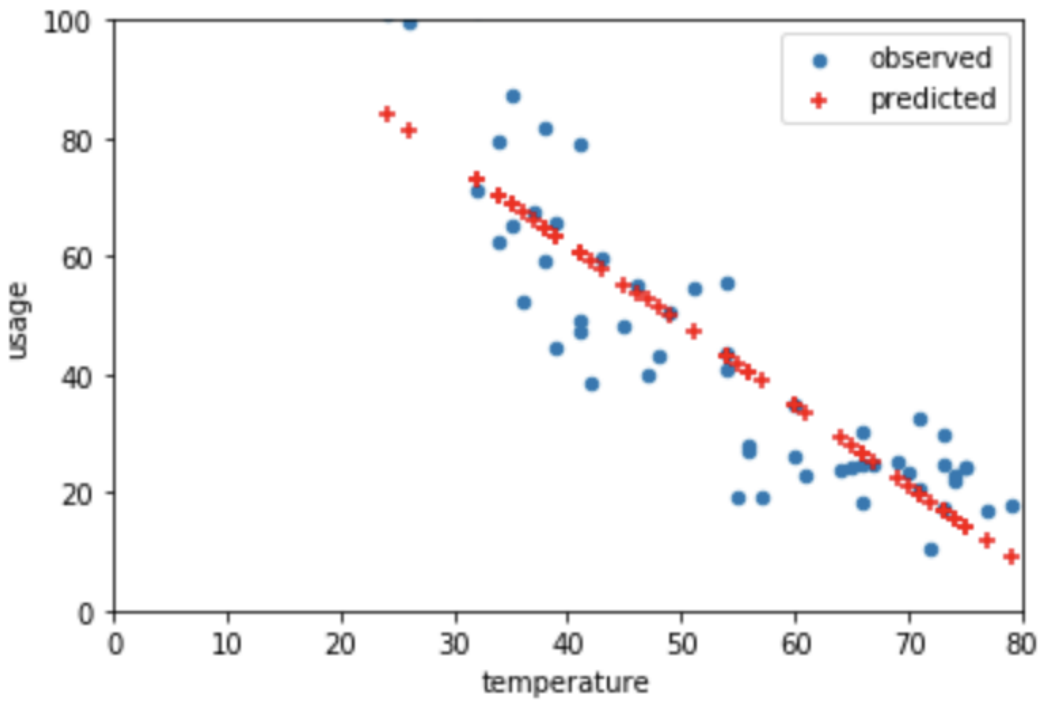
Omnibus:	0.304	**Durbin-Watson:**	1.218
Prob(Omnibus):	0.859	**Jarque-Bera (JB):**	0.474
Skew:	0.131	**Prob(JB):**	0.789
Kurtosis:	2.627	**Cond. No.**	204.

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

Plotting predictions and outcome vs. covariate

```
# plot observed vs predicted usage as a function of temperature
p = usage.plot.scatter(x='temperature', y='usage', label='observed')
p.scatter(x=usage['temperature'], y=Yhat, color='r', marker="+", \
          label='predicted')
plt.legend()
plt.xlim(0,80)
plt.ylim(0,100)
```

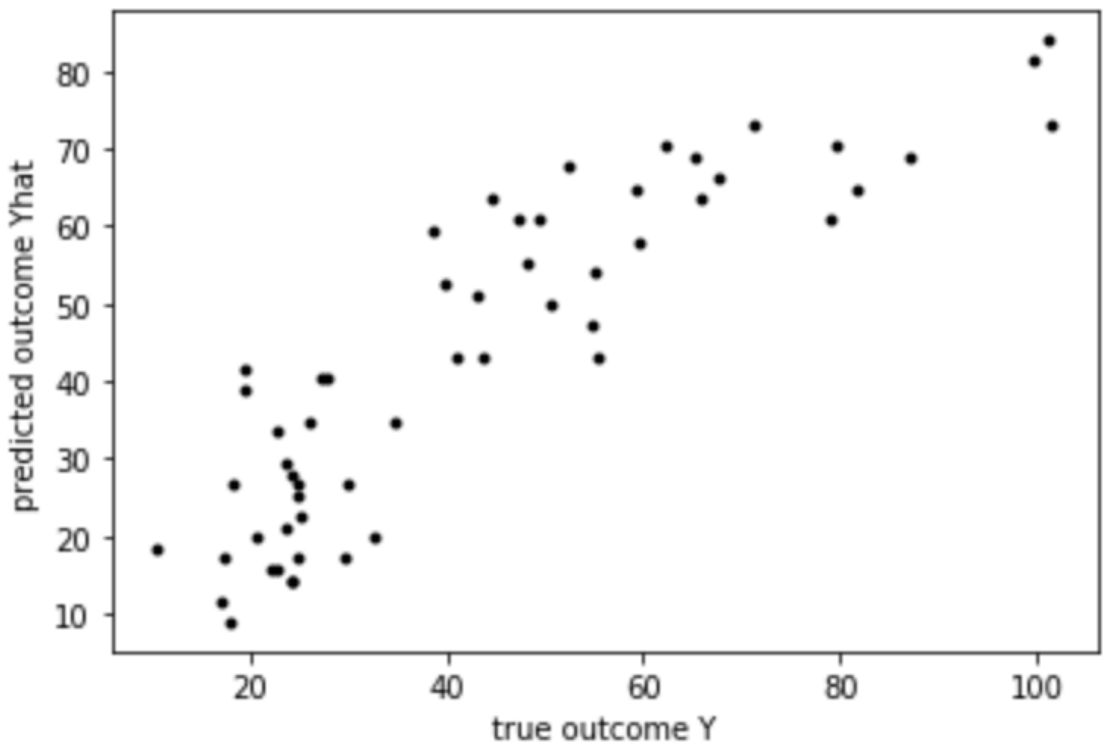
```
(0, 100)
```



Plotting predictions vs. outcome

:
```
# Plot outputs
plt.plot(Y, Yhat, '.', color='black')
plt.xlabel("true outcome Y")
plt.ylabel("predicted outcome Yhat")

plt.show()
```



What if we leave off the intercept?

```
X = usage['temperature']
Y = usage['usage']
model = sm.OLS(Y, X).fit()
Yhat = model.predict(X)
```

```
X.head()
```

```
0    73
1    67
2    57
3    43
4    26
Name: temperature, dtype: int64
```

What if we leave off the intercept?

```
model.summary()
```

OLS Regression Results

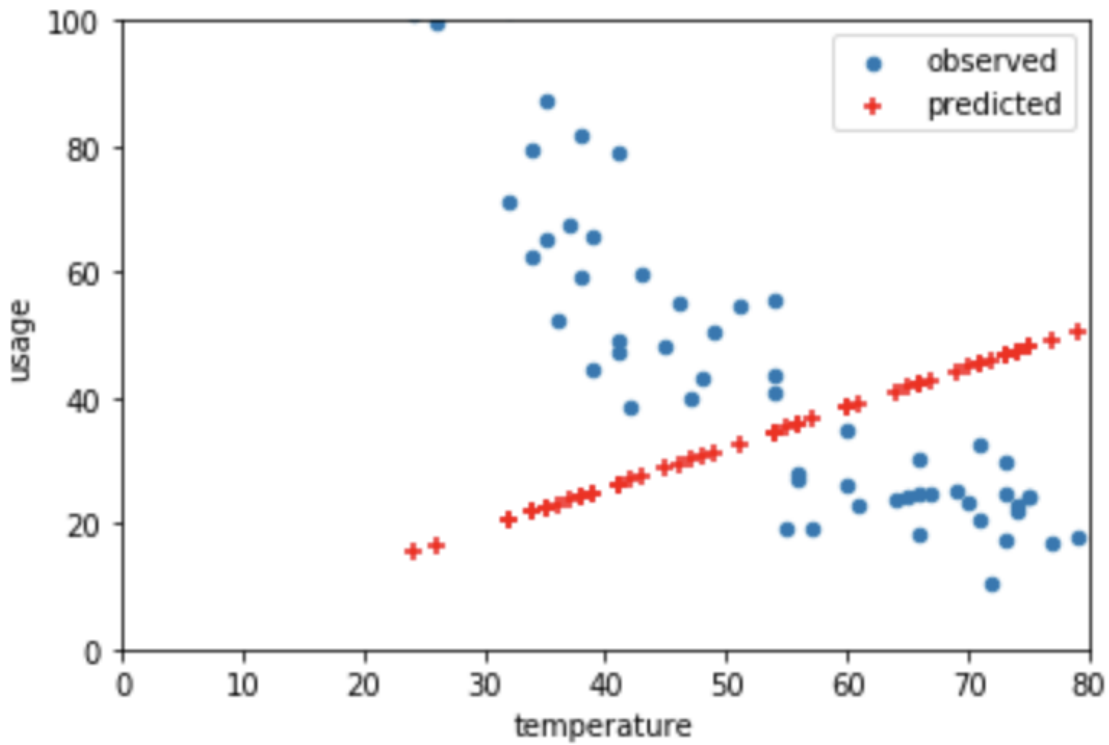
Dep. Variable:	usage	**R-squared (uncentered):**	0.526
Model:	OLS	**Adj. R-squared (uncentered):**	0.517
Method:	Least Squares	**F-statistic:**	59.97
Date:	Fri, 11 Mar 2022	**Prob (F-statistic):**	2.55e-10
Time:	07:33:27	**Log-Likelihood:**	-271.98
No. Observations:	55	**AIC:**	546.0
Df Residuals:	54	**BIC:**	548.0
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
temperature	0.6399	0.083	7.744	0.000	0.474	0.806

Omnibus:	4.990	**Durbin-Watson:**	0.422
Prob(Omnibus):	0.082	**Jarque-Bera (JB):**	4.632
Skew:	0.644	**Prob(JB):**	0.0987
Kurtosis:	2.397	**Cond. No.**	1.00

What if we leave off the intercept?

```
# plot observed vs predicted usage as a function of temperature
p = usage.plot.scatter(x='temperature', y='usage', label='observed')
p.scatter(x=usage['temperature'], y=Yhat, color='r', marker="+", \
     label='predicted')
plt.legend()
plt.xlim(0,80)
plt.ylim(0,100)
```



Here's how to create Y and X using a formula

```
Y, X = dmatrices('usage ~ 1 + temperature',\
                 data=usage, return_type='dataframe')
```

This is the same as

```
Y = usage['usage']
X = usage['temperature']
X = sm.add_constant(X)
```

Formulas: If you leave off the 1, it still creates an intercept

```
Y, X = dmatrices('usage ~ 1 + temperature',\
                 data=usage, return_type='dataframe')
```

is the same as

```
Y, X = dmatrices('usage ~ temperature',\
                 data=usage, return_type='dataframe')
```

Formulas: If you really want to leave off the intercept

If you really want to leave off the intercept, then use 0 instead of 1:

```
Y, X = dmatrices('usage ~ 0 + temperature',\
                 data=usage, return_type='dataframe')
```

:
```
# If you really want to leave off the intercept
Y, X = dmatrices('usage ~ 0 + temperature', \
                 data=usage, return_type='dataframe')
X.head()
```

:

	temperature
0	73.0
1	67.0
2	57.0
3	43.0
4	26.0

Being concise with formulas

First import one additional package

```
import statsmodels.formula.api as smf
```

Now we can fit linear regression models without creating `X` and `Y`

```
model = smf.ols(formula = 'usage ~ 1 + temperature', \
                data=usage).fit()
Yhat = model.predict(exog=usage['temperature'])
```

This is the same as

```
Y, X = dmatrices('usage ~ 1 + temperature',\
                 data=usage, return_type='dataframe')
model = sm.OLS(Y, X).fit()
Yhat = model.predict(X)
```

Let's learn how to read model output

```
# Print out the statistics
model.summary()
```

OLS Regression Results

Dep. Variable:	usage	**R-squared:**	0.780
Model:	OLS	**Adj. R-squared:**	0.776
Method:	Least Squares	**F-statistic:**	188.4
Date:	Wed, 09 Mar 2022	**Prob (F-statistic):**	4.35e-19
Time:	08:26:27	**Log-Likelihood:**	-210.65
No. Observations:	55	**AIC:**	425.3
Df Residuals:	53	**BIC:**	429.3
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	116.7162	5.565	20.973	0.000	105.554	127.878
temperature	-1.3646	0.099	-13.727	0.000	-1.564	-1.165

Omnibus:	0.304	**Durbin-Watson:**	1.218
Prob(Omnibus):	0.859	**Jarque-Bera (JB):**	0.474
Skew:	0.131	**Prob(JB):**	0.789
Kurtosis:	2.627	**Cond. No.**	204.

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

Let's learn how to read the model output

- Estimates, standard errors, p-values and t-values for each β_j
- Residual standard error
- Multiple R-squared and Adjusted R-squared
- F statistic and associated p-value

When are these statistical analyses appropriate?

Statistical quantities in summary (standard errors, p-values, t-values, R squared, F statistic, ...) are computed assuming that $\epsilon_1, \ldots, \epsilon_n$

1. are mutually independent
2. are independent of $X_{i,j}$
3. are normally distributed
4. have a constant variance

- when these assumptions are true, we can use these quantities to lead us towards better models.
- if these assumptions are false, these statistics can be misleading.
- we can alter the data to make assumptions more true; sometimes improves model fit, too!

To check whether these assumptions are true, we must look at the residuals. We will show how to do this later.

Regression coefficients and standard errors

- Estimated regression coefficients: $\hat{\beta}_0, \ldots, \hat{\beta}_p$
- Std. Errors
  - the jth std. error is the standard deviation of $\hat{\beta}_j$
  - an approximate 95% confidence interval for β_j is Estimate $\pm$ (2)(Std. Error)
  - Used to quantify error in the estimates

Statistical tests

we may assess how well the model fits using a variety of statistical tests:

- t value
  - t value = Estimate $\hat{\beta}_j$ / Std. Error of $\hat{\beta}_j$
  - how big is the coefficient's mean relative to its error?
- p value
  - the probability of a t value as large or larger than the one actually observed, if $\beta_j = 0$.
  - If this probability is small, then β_j is probably not 0.
  - Formally, we reject the hypothesis that $\beta_j = 0$ at the $\alpha = 0.05$ significance level if p value < 0.05.

More statistical tests

- Residual std. error
  - estimates the standard deviation of the ϵ_i
- F statistic
  - tests the hypothesis that $\beta_1 = 0, \beta_2 = 0, \ldots,$ and $\beta_p = 0$ i.e., that Y is NOT related to ANY of the predictors
  - often it is obvious that this hypothesis is false
  - used occasionally for more sophisticated procedures built on top of linear regression.

Variance explained

- Multiple R Squared
  - A number between 0 and 1 that tells how well the data is explained by the linear model.
  - Can be used to choose between different predictors.
- Adjusted R Squared
  - Like multiple R squared, but adjusted for the number of predictor variables.
- These are explained on the next few slides.

Multiple R Squared

- Multiple R Squared is the squared correlation between the observed and fitted values,

$$R^2 = \rho(Y, \hat{Y})^2$$

- it is "multiple" because $\hat{Y}$ uses all of the predictors
- $\rho(Y, \hat{Y})$ is the sample correlation

$$\rho(Y, \hat{Y}) = \frac{\sum_i (Y_i - \text{avg}(Y))(\hat{Y}_i - \text{avg}(\hat{Y}))}{\sqrt{\sum_i (Y_i - \text{avg}(Y))^2} \sqrt{\sum_i (\hat{Y}_i - \text{avg}(\hat{Y}))^2}}$$

- The closer R^2 is to 1, the better $\hat{Y}_i$ predicts Y_i.

Why use R^2?

- R^2 can be used to determine which set of predictors is best.
  - bigger R^2 is better
- problem: R^2 is biased in favor of more predictors
  - adding predictors increases R^2
  - even if the additional predictors are not related to Y_i
- The bias of R^2 can be fixed by using

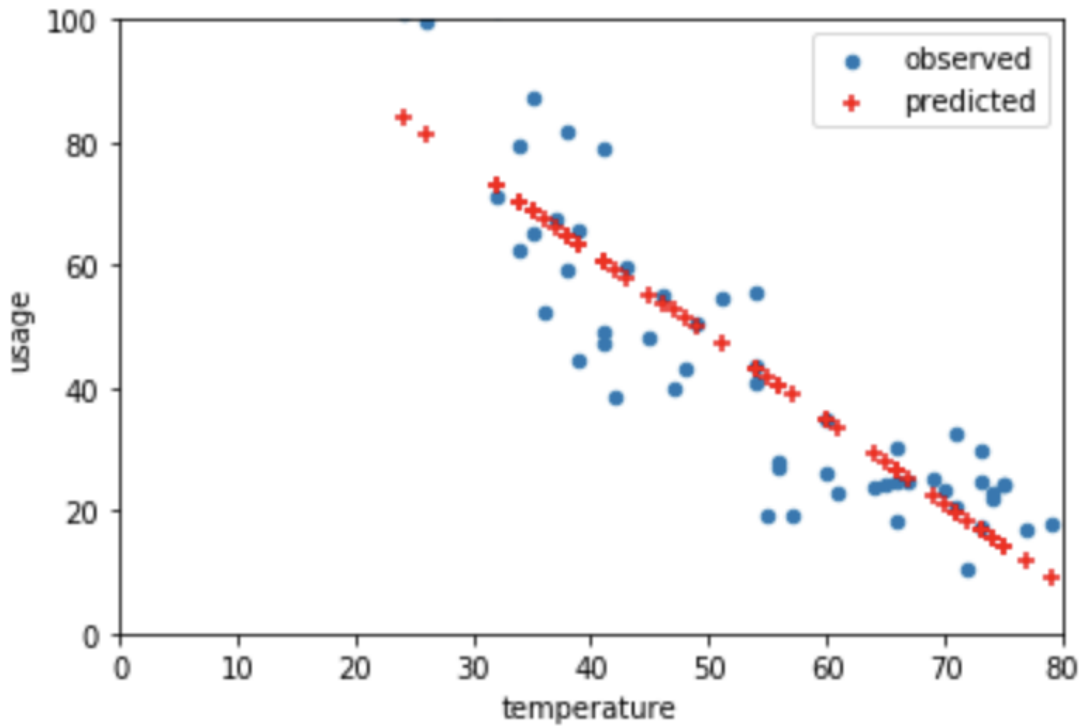
$$\text{Adjusted } R^2 = 1 - (1 - R^2)\frac{n-1}{n-p-1}$$

  (recall n=number of observations, p=number of covariates)
- Adjusted R^2 is preferred to R^2

Back to the demo: Motivation for adding temperature squared as a covariate

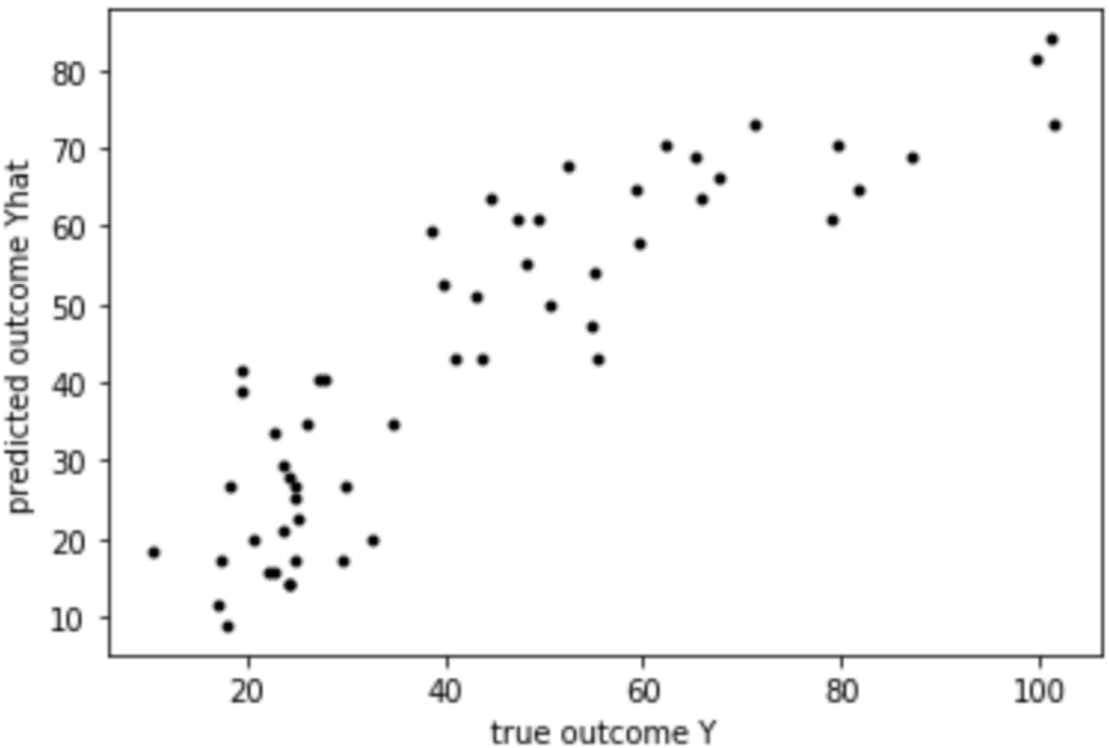
Our original model had an R^2 of 0.780 and an adjusted R^2 of 0.776. Can we do better?

Plotting our outcome vs. our covariate, our predictions are straight but our data is curved.



Motivation for adding temperature squared as a covariate

We also see curvature when plotting outcomes vs. predictions



Adding temperature squared as a covariate

We might predict better if we add a covariate that curves with temperature

$$\texttt{usage} = \beta_0 + \beta_1 \texttt{temp} + \beta_2 \texttt{temp}^2 + \epsilon$$

```
Y, X = dmatrices('usage ~ 1 + temperature + np.power(temperat
                  data=usage, return_type='dataframe')
model = sm.OLS(Y, X).fit()
Yhat = model.predict(X)
```

```
X.head()
```

	Intercept	temperature	np.power(temperature, 2)
0	1.0	73.0	5329.0
1	1.0	67.0	4489.0
2	1.0	57.0	3249.0
3	1.0	43.0	1849.0
4	1.0	26.0	676.0

Here's our new model output

```
# Print out the statistics
model.summary()
```

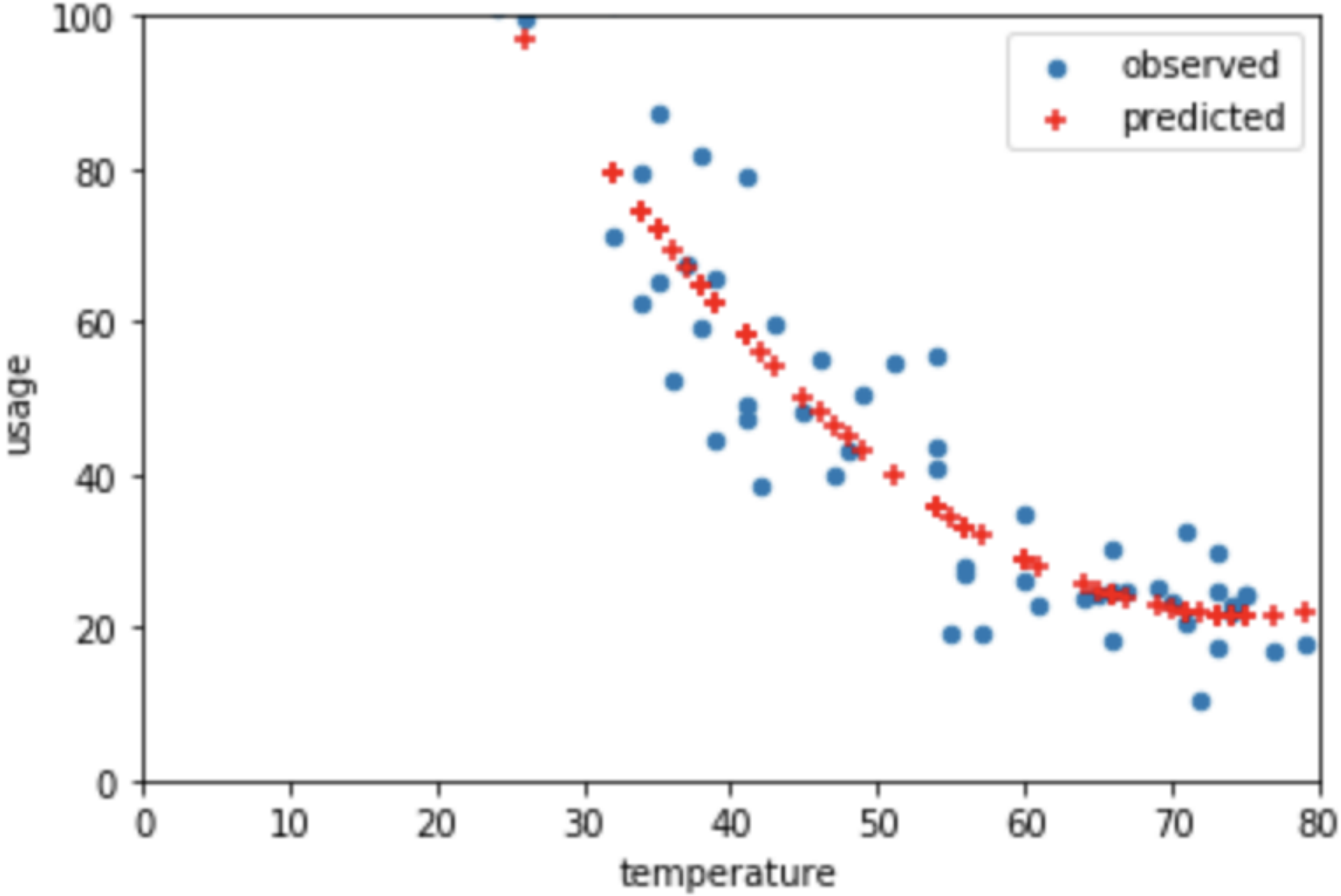
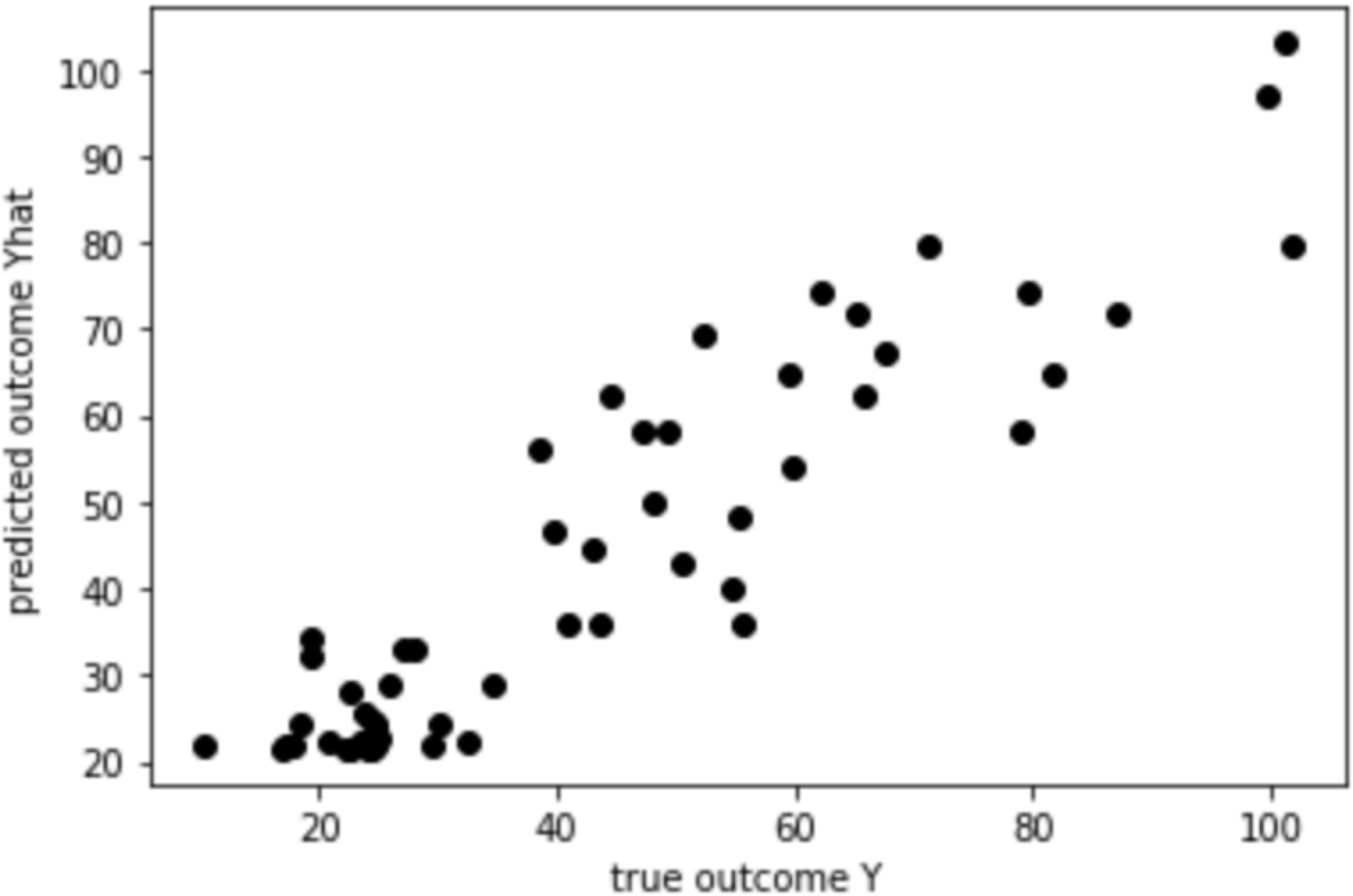
OLS Regression Results

Dep. Variable:	usage	**R-squared:**	0.847
Model:	OLS	**Adj. R-squared:**	0.841
Method:	Least Squares	**F-statistic:**	143.8
Date:	Wed, 09 Mar 2022	**Prob (F-statistic):**	6.47e-22
Time:	08:27:55	**Log-Likelihood:**	-200.75
No. Observations:	55	**AIC:**	407.5
Df Residuals:	52	**BIC:**	413.5
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	196.7153	17.488	11.249	0.000	161.624	231.807
temperature	-4.6405	0.695	-6.678	0.000	-6.035	-3.246
np.power(temperature, 2)	0.0307	0.006	4.749	0.000	0.018	0.044

Omnibus:	1.155	**Durbin-Watson:**	1.166
Prob(Omnibus):	0.561	**Jarque-Bera (JB):**	0.888
Skew:	0.310	**Prob(JB):**	0.642
Kurtosis:	2.955	**Cond. No.**	4.80e+04

Here's our new plot of outcomes vs. temperature



Here's our new plot of outcomes vs. predictions