GENERAL ASSEMBLY

INTRODUCTION TO REGRESSION ANALYSIS

Adam Jones, PhD

Data Scientist @ Critical Juncture

LEARNING OBJECTIVES

- Define data modeling and simple linear regression
- Build a linear regression with data that meets the linearity assumption using the sci-kit learn library
- Understand and identify multicollinearity in multiple regression

INTRODUCTION TO REGRESSION ANALYSIS

PRE-WORK

PRE-WORK REVIEW

- Effectively show *correlations* between an independent variable x and a dependent variable y
- Be familiar with the *get_dummies* function in pandas
- Understand the difference between *vectors, matrices, Series,* and *DataFrames*
- Understand the concepts of *outliers* and *distance*
- Be able to interpret *p values* and *confidence intervals*

OPENING

INTRODUCTION TO REGRESSION ANALYSIS

WHERE ARE WE IN THE DATA SCIENCE WORKFLOW?

- Data has been **acquired** and **parsed**
- Today we’ll **refine** the data and **build** models
- We’ll also use plots to **represent** the results

INTRODUCTION

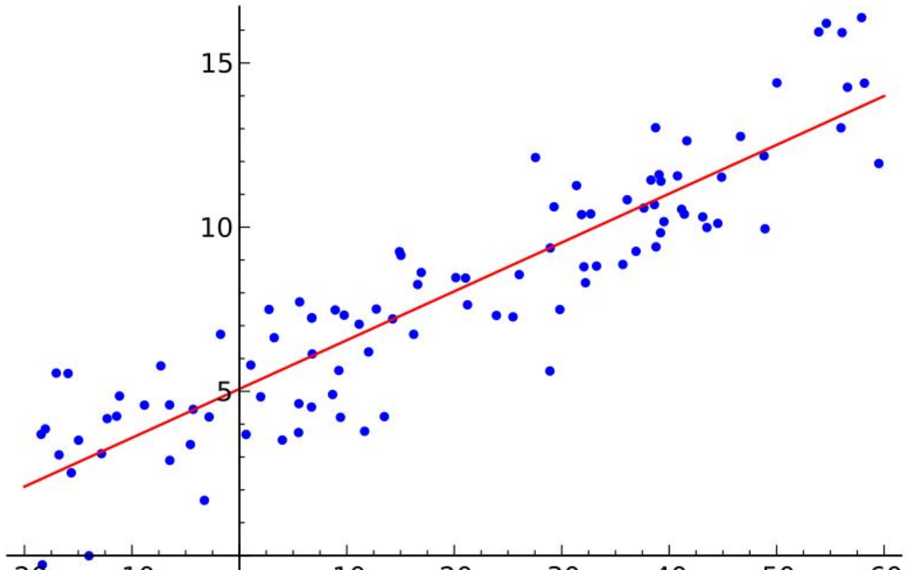
SIMPLE LINEAR REGRESSION

SIMPLE LINEAR REGRESSION

- **Simple linear regression** is a *statistical method* that allows us to summarize and study relationships between two continuous (quantitative) variables

- The simplest version is just a line of best fit:

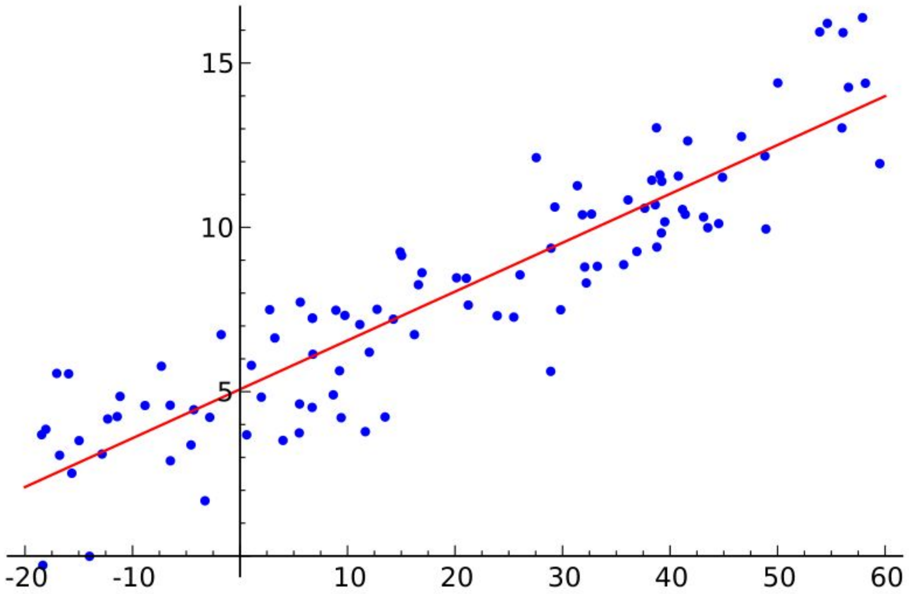
$$y = mx + b$$



- Explain the relationship between **x** and **y** using the starting point **b** and the power in explanation **m**

SIMPLE LINEAR REGRESSION

- **Simple linear regression** is a *statistical method* that allows us to summarize and study relationships between two continuous (quantitative) variables



- The simplest version is just a line of best fit:

$$y = mx + b$$

- Explain the relationship between **x** and **y** using the starting point **b** and the power in explanation **m**

MULTIPLE LINEAR REGRESSION

- However, *multiple* linear regression uses linear algebra to explain the relationship between *multiple* x's and y
- The more sophisticated version: y = beta * X + alpha
- Explain the relationship between the matrix **X** and a dependent vector **y** using a y-intercept **alpha** and the relative coefficients **beta**

LINEAR REGRESSION

- Linear regression works **best** when:
  - The data is normally distributed
  - X’s significantly explain y
  - X’s are independent of each other
  - Resulting values pass linear assumption
- If data is not normally distributed, we could introduce *bias*

DEMO

REGRESSING AND NORMAL DISTRIBUTIONS

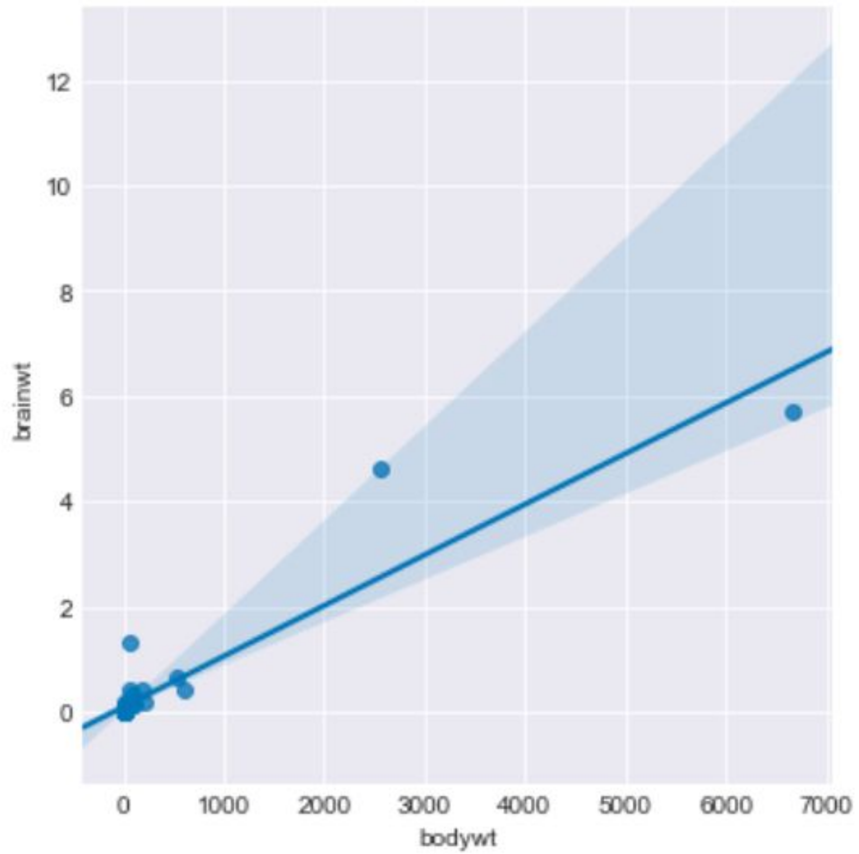
DEMO: REGRESSING AND NORMAL DISTRIBUTIONS

- The first plot shows a relationship between two values
  - though not a linear solution

- Note that lmplot() returns a straight line plot

```
In [6]: sns.lmplot('bodywt', 'brainwt', mammals)

Out[6]: <seaborn.axisgrid.FacetGrid at 0x1a09b75850>
```

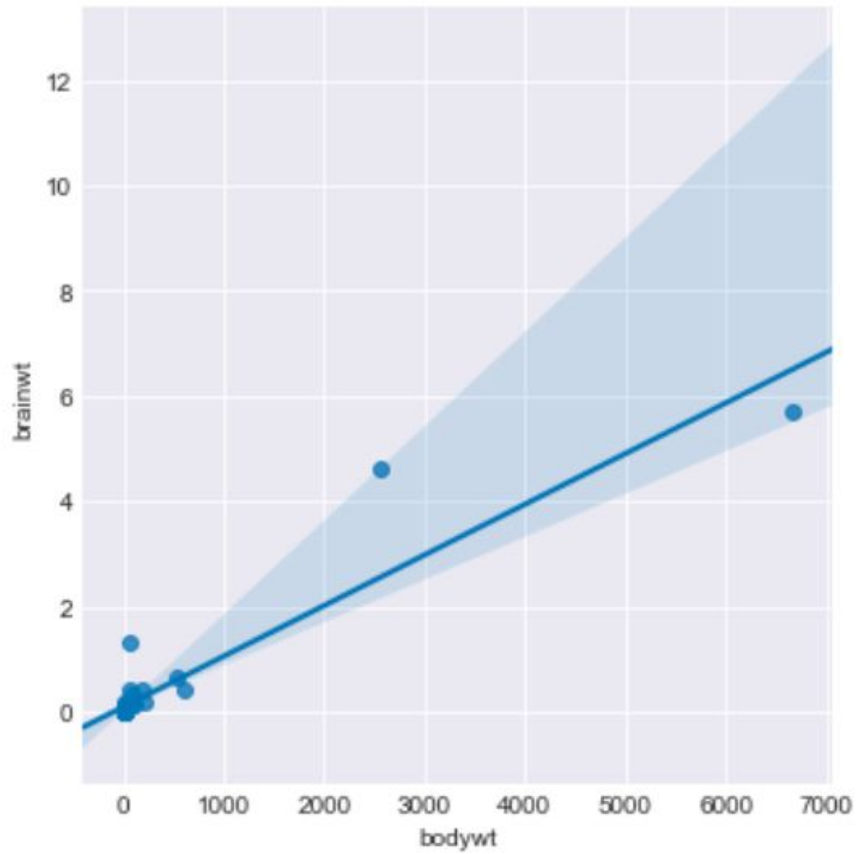


DEMO: REGRESSING AND NORMAL DISTRIBUTIONS

- The first plot shows a relationship between two values
  - though not a linear solution

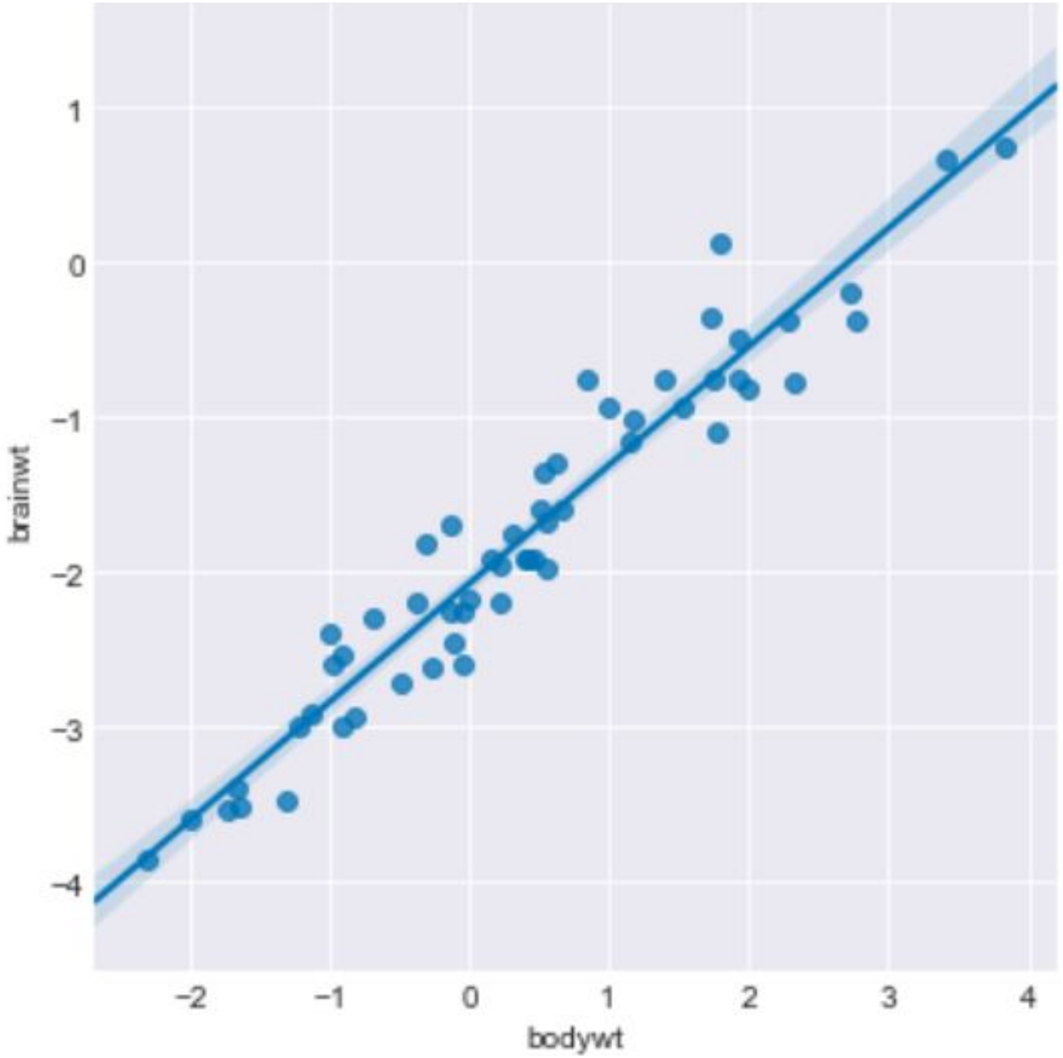
- Note that lmplot() returns a straight line plot

```
In [6]: sns.lmplot('bodywt', 'brainwt', mammals)
Out[6]: <seaborn.axisgrid.FacetGrid at 0x1a09b75850>
```



DEMO: REGRESSING AND NORMAL DISTRIBUTIONS

- However, we can transform the data, both log-log distributions to get a linear solution



GUIDED PRACTICE

USING SEABORN TO GENERATE SIMPLE LINEAR MODEL PLOTS

ACTIVITY: GENERATE SINGLE VARIABLE LINEAR MODEL PLOTS

EXERCISE

DIRECTIONS (15 minutes)

1. Update and complete the code in the starter notebook to use **lmplot** and display correlations between body weight and two dependent variables: **sleep_rem** and **awake**

DELIVERABLE

Two plots

INTRODUCTION

SIMPLE REGRESSION ANALYSIS IN SKLEARN

CLASSES AND OBJECTS IN OBJECT ORIENTED PROGRAMMING

- **Classes** are an abstraction for a complex set of ideas, e.g. *human*

- Specific **instances** of classes can be created as **objects**
  - *john_smith = human()*

- Objects have **properties** (i.e. attributes or other information)
  - *john_smith.age*
  - *john_smith.gender*

- Object have **methods** (i.e. procedures associated with a class/object)
  - *john_smith.breath()*
  - *john_smith.walk()*

SIMPLE LINEAR REGRESSION ANALYSIS IN SKLEARN

- Sklearn defines models as *objects* (in the OOP sense)
- You can use the following principles:
  - All sklearn modeling classes are based on the base estimator
    - This means **all** models take a similar form
  - All estimators take a matrix **X**
  - Supervised estimators also take a vector **y** (the response)
  - Estimators can be customized through setting the appropriate parameters

SIMPLE LINEAR REGRESSION ANALYSIS IN SKLEARN

- General format for sklearn model classes and methods

```
# generate an instance of an estimator class
estimator = base_models.AnySKLearnObject()
# fit your data
estimator.fit(X, y)
# score it with the default scoring method (recommended to use the metrics module in the future)
estimator.score(X, y)
# predict a new set of data
estimator.predict(new_X)
# transform a new X if changes were made to the original X while fitting
estimator.transform(new_X)
```

- LinearRegression() doesn’t have a transform function

- With this, we can build a simple process for linear regression

DEMO

SIGNIFICANCE IS KEY

DEMO: SIGNIFICANCE IS KEY

- What does the residual plot tell us?

- How can we use the linear assumption?

DEMO: SIGNIFICANCE IS KEY

- What does the residual plot tell us?

- How can we use the linear assumption?

```
X = mammals[['bodywt']]
y = mammals['brainwt']
lm = linear_model.LinearRegression()
lm = get_linear_model_metrics(X, y, lm)
```

```
P Values: [9.15540205e-26]
Coefficients: [0.00096395]
y-intercept: 0.0859173102936466
R-Squared: 0.8719491980865914
```

DEMO: SIGNIFICANCE IS KEY

- What does the residual plot tell us?

- How can we use the linear assumption?

```
lm = linear_model.LinearRegression(fit_intercept=False)
lm = get_linear_model_metrics(X, y, lm)
# prediction at 0?
print lm.predict([[0]])
```

```
P Values: [9.15540205e-26]
Coefficients: [0.00098291]
y-intercept: 0.0
R-Squared: 0.864418807451033
[0.]
```

GUIDED PRACTICE

USING THE LINEAR REGRESSION OBJECT

ACTIVITY: USING THE LINEAR REGRESSION OBJECT

EXERCISE

DIRECTIONS (15 minutes)

1. Generate two more models using the log-transformed data to see how this transform changes the model's performance
2. Use the code on the following slide to complete #1

DELIVERABLE

Two new models

ACTIVITY: USING THE LINEAR REGRESSION OBJECT

EXERCISE

DIRECTIONS (15 minutes)

```
X =
y =
loop = []
for boolean in loop:
    print 'y-intercept:', boolean
    lm =
linear_model.LinearRegression(fit_intercept=boolean)
    get_linear_model_metrics(X, y, lm)
    print
```

DELIVERABLE

Two new models

INDEPENDENT PRACTICE

BASE LINEAR REGRESSION CLASSES

ACTIVITY: BASE LINEAR REGRESSION CLASSES

EXERCISE

DIRECTIONS (20 minutes)

1. Experiment with the model evaluation function we have (**get_linear_model_metrics**) with the following sklearn estimator classes

    a. linear_model.Lasso()
    b. linear_model.Ridge()
    c. linear_model.ElasticNet()

Note: We'll cover these new regression techniques in a later class

DELIVERABLE

New models and evaluation metrics

INTRODUCTION

MULTIPLE REGRESSION ANALYSIS

MULTIPLE REGRESSION ANALYSIS

- Simple linear regression with one variable can explain some variance, but using multiple variables can be much more powerful
- We want our multiple variables to be mostly independent to avoid multicollinearity
- Multicollinearity, when two or more variables in a regression are highly correlated, can cause problems with the model

MULTIPLE REGRESSION ANALYSIS

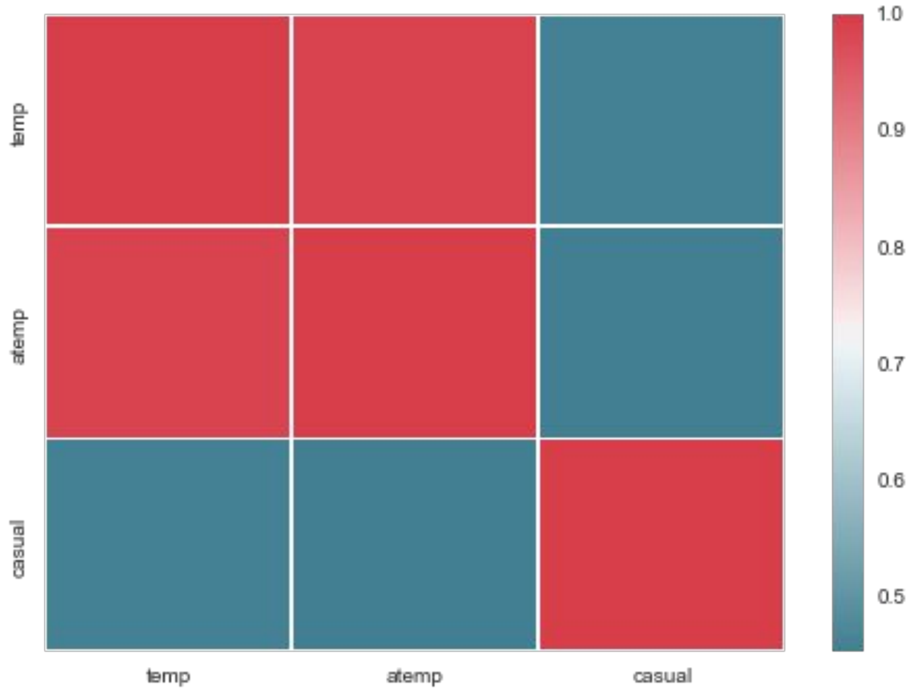
- Simple linear regression with one variable can explain some variance, but using multiple variables can be much more powerful
- We want our multiple variables to be mostly independent to avoid multicollinearity
- Multicollinearity, when two or more variables in a regression are highly correlated, can cause problems with the model

BIKE DATA EXAMPLE

- We can look at a *correlation matrix* of our bike data

- Even if adding correlated variables to the model improves overall variance, it can introduce problems when explaining the output of your model

- What happens if we use a second variable that isn't highly correlated with temperature?



BIKE DATA EXAMPLE

- We can look at a *correlation matrix* of our bike data

- Even if adding correlated variables to the model improves overall variance, it can introduce problems when explaining the output of your model

- What happens if we use a second variable that isn't highly correlated with temperature?

```
temp
P Values: [0.]
Coefficients: [117.68705779]
y-intercept: -22.812739187991426
R-Squared: 0.2112465416296054
()
atemp
P Values: [0.]
Coefficients: [130.27875081]
y-intercept: -26.307167548054544
R-Squared: 0.20618870573273862
()
temp, atemp
P Values: [0. 0.]
Coefficients: [116.34021588   1.52795677]
y-intercept: -22.87033982864664
R-Squared: 0.21124723660961908
()
```

BIKE DATA EXAMPLE

- We can look at a *correlation matrix* of our bike data

- Even if adding correlated variables to the model improves overall variance, it can introduce problems when explaining the output of your model

- What happens if we use a second variable that isn't highly correlated with temperature?

```
P Values: [0. 0.]
Coefficients: [112.02457031 -80.87301833]
y-intercept: 30.727333858091797
R-Squared: 0.3109011969133708
```

GUIDED PRACTICE

MULTICOLLINEARITY WITH DUMMY VARIABLES

ACTIVITY: MULTICOLLINEARITY WITH DUMMY VARIABLES

EXERCISE

DIRECTIONS (15 minutes)

1. Load the bike data
2. Run through the code on the following slide
3. What happens to the coefficients when you include all weather situations instead of just including all except one?

DELIVERABLE

Two models' output

ACTIVITY: MULTICOLLINEARITY WITH DUMMY VARIABLES

EXERCISE

DIRECTIONS (15 minutes)

```
lm = linear_model.LinearRegression()
weather = pd.get_dummies(bike_data.weathersit)
get_linear_model_metrics(weather[[1, 2, 3, 4]], y, lm)
print
# drop the least significant, weather situation  = 4
get_linear_model_metrics(weather[[1, 2, 3]], y, lm)
```

DELIVERABLE

Two models' output

GUIDED PRACTICE

COMBINING FEATURES INTO A BETTER MODEL

ACTIVITY: COMBINING FEATURES INTO A BETTER MODEL

EXERCISE

DIRECTIONS (15 minutes)

1. Complete the code on the following slide (partner optional)
2. Visualize the correlations of all the numerical features built into the dataset
3. Add the three significant weather situations into our current model
4. Find two more features that are not correlated with the current features, but could be strong indicators for predicting guest riders

DELIVERABLE

Visualization of correlations, new models

INDEPENDENT PRACTICE

BUILDING MODELS FOR OTHER Y VARIABLES

ACTIVITY: COMBINING FEATURES INTO A BETTER MODEL

EXERCISE

DIRECTIONS (15 minutes)

```
lm = linear_model.LinearRegression()
bikemodel_data = bike_data.join() # add in the three weather situations

cmap = sns.diverging_palette(220, 10, as_cmap=True)
correlations = # what are we getting the correlations of?
print correlations
print sns.heatmap(correlations, cmap=cmap)

columns_to_keep = [] #[which_variables?]
final_feature_set = bikemodel_data[columns_to_keep]

get_linear_model_metrics(final_feature_set, y, lm)
```

DELIVERABLE

Visualization of correlations, new models

ACTIVITY: BUILDING MODELS FOR OTHER Y VARIABLES

EXERCISE

DIRECTIONS (25 minutes)

1. Build a new model using a new y variable: registered riders.
2. Pay attention to the following:
   a. the distribution of riders (should we rescale the data?)
   b. checking correlations between the variables and y variable
   c. choosing features to avoid multicollinearity
   d. model complexity vs. explanation of variance
   e. the linear assumption

BONUS

1. Which variables make sense to dummy?
2. What features might explain ridership but aren't included? Can you build these features with the included data and pandas?

DELIVERABLE

A new model and evaluation metrics

ACTIVITY: BUILDING MODELS FOR OTHER Y VARIABLES

EXERCISE

DIRECTIONS (25 minutes)

1. Build a new model using a new y variable: registered riders.
2. Pay attention to the following:
    a. the distribution of riders (should we rescale the data?)
    b. checking correlations between the variables and y variable
    c. choosing features to avoid multicollinearity
    d. model complexity vs. explanation of variance
    e. the linear assumption

BONUS

1. Which variables make sense to dummy?
2. What features might explain ridership but aren't included? Can you build these features with the included data and pandas?

DELIVERABLE

A new model and evaluation metrics

CONCLUSION

TOPIC REVIEW

CONCLUSION

- You should now be able to answer the following questions:
    - What is simple linear regression?
    - What makes multi-variable regressions more useful?
    - What challenges do they introduce?
    - How do you dummy a category variable?

WEEK 3 : LESSON 6

UPCOMING WORK

UPCOMING WORK

Week 4 : Lesson 8

- Project: Final Project, Deliverable 1- DUE Thurs (4/12)!

Q & A

EXIT TICKET

DON'T FORGET TO FILL OUT YOUR EXIT TICKET!