GENERAL ASSEMBLY

# INTRODUCTION TO REGRESSION ANALYSIS

*Naumaan Nayyar*

# LEARNING OBJECTIVES

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

# VARIABLE TYPES

## VARIABLE TYPES

- Numeric variables can take on a large range of non-predetermined, quantitative values. These are things such as height, income, etc.

- Categorical variables can take on a specific set of variables.  These are things such as race, gender, paint colors, movie titles, etc.

DEMO

# CLASSES

## CLASS/DUMMY VARIABLES

- Let's say we have the categorical variable area, which takes on one of the following values: rural, suburban, and urban.

- We need to represent these numerically for a model. So how do we code them?

## CLASS/DUMMY VARIABLES

- How about `0=rural`, `1=suburban`, and `2=urban`?

## CLASS/DUMMY VARIABLES

- But this implies an ordered relationship - is `urban` twice `suburban`? That doesn't make sense.

- However, we can represent this information by converting the one `area` variable into two new variables:

  `area_urban` and `area_suburban`.

## CLASS/DUMMY VARIABLES

- We'll draw out how categorical variables can be represented without implying order.
- First, let's choose a reference category.  This will be our "base" category.
- It's often good to choose the category with the largest sample size and a criteria that will help model interpretation.  If we are testing for a disease, the reference category would be people without the disease.

# CLASS/DUMMY VARIABLES

- Step 1: Select a reference category. We'll choose `rural` as our reference category.
- Step 2: Convert the values `urban`, `suburban`, and `urban` into a numeric representation that does not imply order.
- Step 3: Create two new variables: `area_urban` and `area_suburban`.

## CLASS/DUMMY VARIABLES

- Why do we need only two dummy variables?

| | | |
|---|---|---|
| `rural` | `urban` | `suburban` |

- We can derive all of the possible values from these two.  If an `area` isn't `urban` or `suburban`, we know it must be `rural`.

- In general, if you have a categorical feature with k categories, you need to create k-1 dummy variable to represent all of the information.

# CLASS/DUMMY VARIABLES

- Let's see our dummy variables.

| | **area_urban** | **area_suburban** |
|---|---|---|
| rural | 0 | 0 |
| suburban | 0 | 1 |
| urban | 1 | 0 |

- As mentioned before, if we know `area_urban=0` and `area_suburban=0`, then the area must be `rural`.

## CLASS/DUMMY VARIABLES

- We can do this for a `gender` variable with two categories: `male` and `female`.
- How many dummy variables need to be created?

## CLASS/DUMMY VARIABLES

- # of categories - 1 = 2 -1 = 1

## CLASS/DUMMY VARIABLES

- We will make `female` our reference category. Thus, `female=0` and `male=1`.

| | **gender_male** |
|---|---|
| female | 0 |
| male | 1 |

- This can be done in Pandas with the `get_dummies` method.

INDEPENDENT PRACTICE

# DUMMY COLORS

# ACTIVITY: DUMMY COLORS

EXERCISE

## DIRECTIONS (15 minutes)

It's important to understand the concept before we use the Pandas function get_dummies to create dummy variables.  So today, we'll create our dummy variables by hand.

1. Draw a table like the one on the white board.
2. Create dummy variables for the variable "colors" that has 6 categories:  blue, red, green, purple, grey, and brown.  Use grey as the reference.

## DELIVERABLE

Dummy variables table for colors

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.

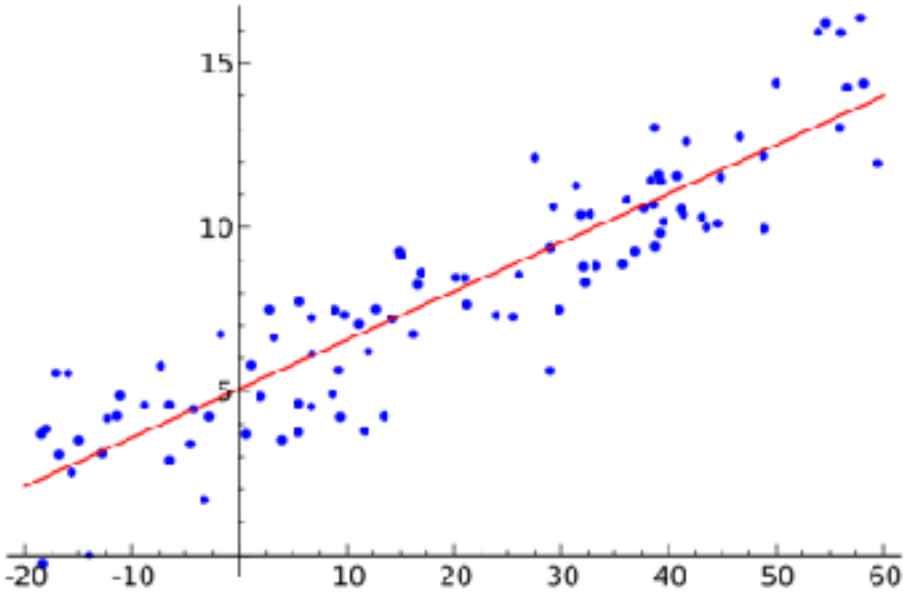
# SIMPLE LINEAR REGRESSION

# SIMPLE LINEAR REGRESSION

- Def: Explanation of a continuous variable given a series of independent 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

- However, linear regression uses linear algebra to explain the relationship between *multiple* x's and y.

- The more sophisticated version:  y = beta * X + alpha (+ error)

- Explain the relationship between the matrix **X** and a dependent vector **y** using a y-intercept **alpha** and the relative coefficients **beta**.

## SIMPLE LINEAR REGRESSION

- Linear regression works **best** when:
  - The data is normally distributed (but doesn't have to be)
  - X's significantly explain y (have low p-values)
  - X's are independent of each other (low multicollinearity)
  - Resulting values pass linear assumption (depends upon problem)
- If data is not normally distributed, we could introduce *bias*.

DEMO

# REGRESSING AND NORMAL DISTRIBUTIONS

## DEMO: REGRESSING AND NORMAL DISTRIBUTIONS

- Follow along with your starter code notebook while I walk through these examples.
- The first plot shows a relationship between two values, though not a linear solution.
- Note that lmplot() returns a straight line plot.
- 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

## 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**, either sparse or dense.
    - Supervised estimators also take a vector **y** (the response).
    - Estimators can be customized through setting the appropriate parameters.

## 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**.  These are attributes or other information.
  - *john_smith.age*
  - *john_smith.gender*

- Object have **methods**.  These are procedures associated with a class/object.
  - *john_smith.breathe()*
  - *john_smith.walk()*

# 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 information, we can build a simple process for linear regression.

DEMO

# SIGNIFICANCE IS KEY

## DEMO: SIGNIFICANCE IS KEY

- Follow along with your starter code notebook while I walk through these examples.

- What does the residual plot tell us?

- How can we use the linear assumption?

## GUIDED PRACTICE

# USING THE LINEAR REGRESSION OBJECT

# ACTIVITY: USING THE LINEAR REGRESSION OBJECT

EXERCISE

## DIRECTIONS (15 minutes)

1. With a partner, 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.

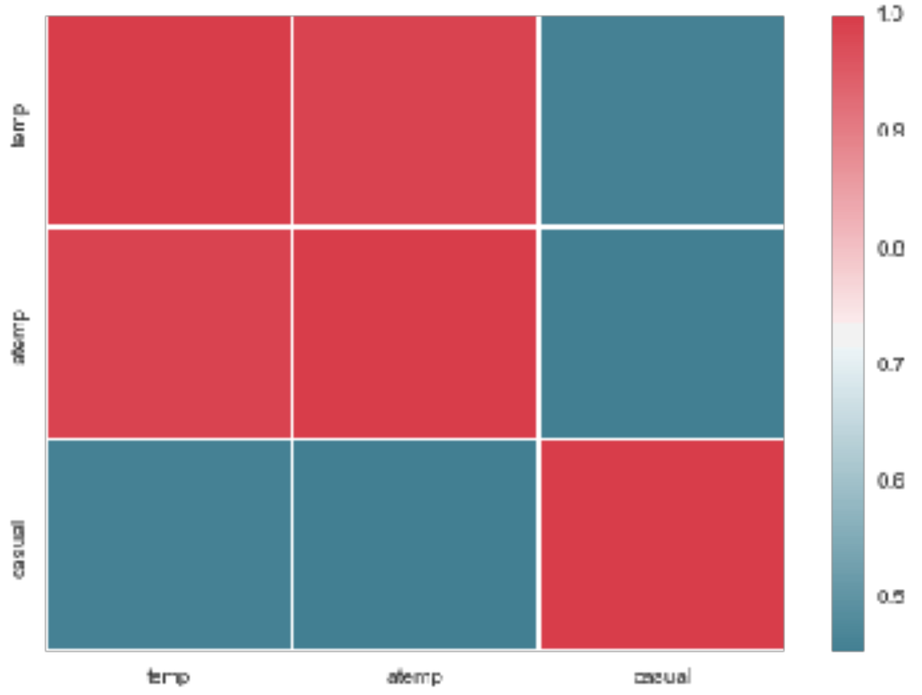
- Multicollinearity, when two or more variables in a regression are highly correlated, can cause problems with the model.

# 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?



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. With a partner, complete the code on the following slide.
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

# 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

## INDEPENDENT PRACTICE

# BUILDING MODELS FOR OTHER Y VARIABLES

# 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?
  - How do you avoid a singular matrix?

# UPCOMING WORK

## UPCOMING WORK

# Week 4 : Lesson 7

- Project: Unit Project 2

# Q & A

# EXIT TICKET

DON’T FORGET TO FILL OUT YOUR EXIT TICKET!