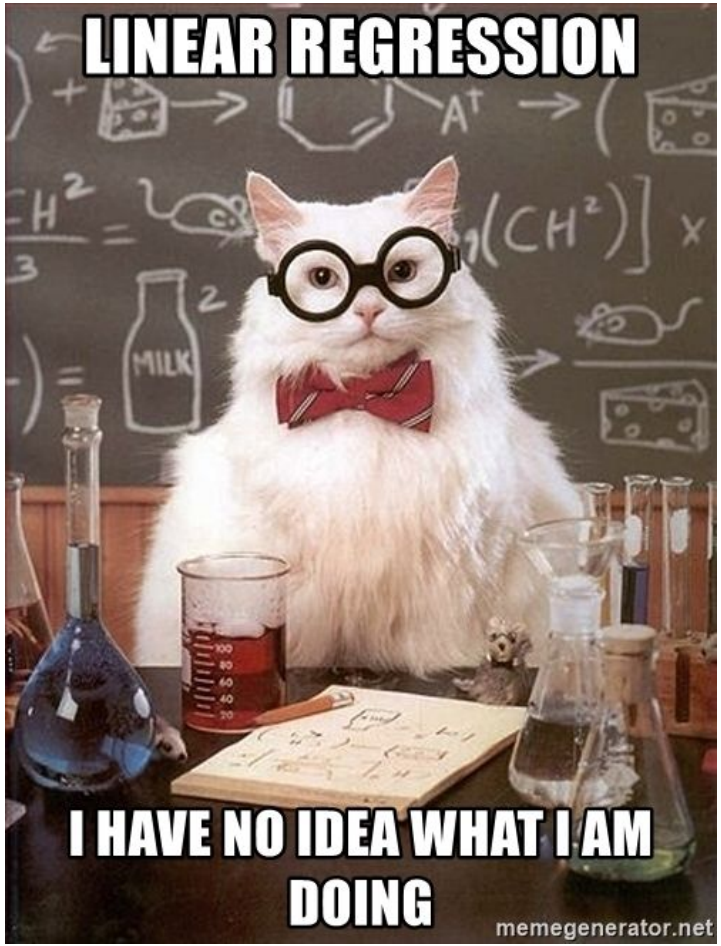


Image Source

Image Source

Welcome to Week 16, Lecture 02!

Diagnosing and Interpreting Linear Regression

Agenda

- Assignments
- Announcements

- Last Class:
  - Why Use Linear Regression?
  - How does Linear Regression work?
  - What are coefficients?
  - The 4 Assumptions of Linear Regression
  - Demo: How to fit a linear regression with statsmodels.

- Today:
  - Demo: How to fit a linear regression with statsmodels.
  - Diagnosing a Regression model
  - Iterating on our model
  - Advanced approaches for multicollinearity
  - Coefficient Interpretation

Assignments

Only 1 required assignment:

- **Project 3 - Part 4(Core)**

Highly Recommended:

- Do Project 3 - Part 5 for your portfolio!!!

Week 3 assignment feedback added!

Final Assignment Deadline = Friday at 9 AM PST!

- All resubmissions from week 1 and 2 .
- All week 3 and 4 assignments turned in.
- **Grace Period for Week 3 Resubmits:**
  - If you are asked for resubmissions for week 3 assignments, the deadline for submitting those is Monday at 9 AM PST.

📣 Announcements

- 🙏 **Please take a few minutes to fill out the End of Stack survey.** 🥺
  https://login.codingdojo.com/dashboard/end-stack-survey

- 🎓 **Online DS Graduation!**
  - **Friday, May 27th @ 5 PM PST**
    - Our cohort + students from 12 week program.
    - In the same Zoom Room as lecture

- **Bonus Lecture Tomorrow:**
  - **Topic: Creating Your Own Python Package/Module**
  - Date: Friday 05/13/22 @ 5 PM PST
    - We will start with a review of writing functions
    - Then discuss moving our functions to an external .py file (making a module)
    - Then we will discuss how to make an actual PyPi Package (pip-installable)

Linear Regression Assumptions

The 4 Assumptions of Linear Regression

The first 2 assumptions are about the **features:**

- **Linearity**
- **Independence of features** (AKA Little-to-No Multicollinearity)

The last 2 assumptions are about the **residuals (errors):**

- **Normality**
- **Homoscedasticity**

Fitting a Linear Regression with Statsmodels

Linear Regression Modeling is Iterative

- Be prepared to try MANY versions of your model with changes to your features.
- Functionize your data prep and evaluation process for easy iteration.
- Keep your notebook organized with headers!
  - Add a header for each iteration of your model.

Walkthrough: Linear Regression with statsmodels

Fitting a linear regression to predict movie revenue.

- Basically, the optional assignment: Project 3 - Part 5.
- Repo: https://github.com/coding-dojo-data-science/data-enrichment-linear-regression-with-movies

- Last Class:
  - Preparing the data for a statsmodels OLS.
  - Fitting an OLS model.

- Next Class:
  - Interpreting the Model Summary and its coefficients.
  - Diagnosing the model
  - Better meeting the assumptions
  - Advanced approaches to dealing with multicollinearity

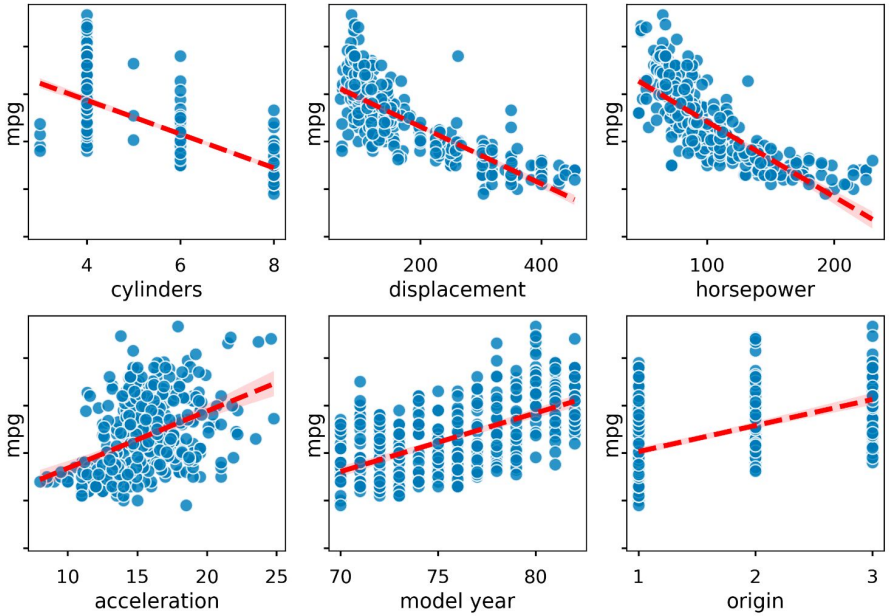
- BUT BEFORE WE DIVE BACK IN….

Diagnosing Linear Regression Models

Assumption of Linearity

That the input features have a linear relationship with the target.

- To check:
  - Use visualizations!



If your features violate linearity

- Simplest solution:
  - Drop features that are not linearly correlated with target.

- More Advanced Solutions (not covered today):
  - Transform your non-linear features into PolynomialFeatures.
    - Towards Data Science: Polynomial Regression
    - Machine Learning Mastery: Polynomial Feature Transformations

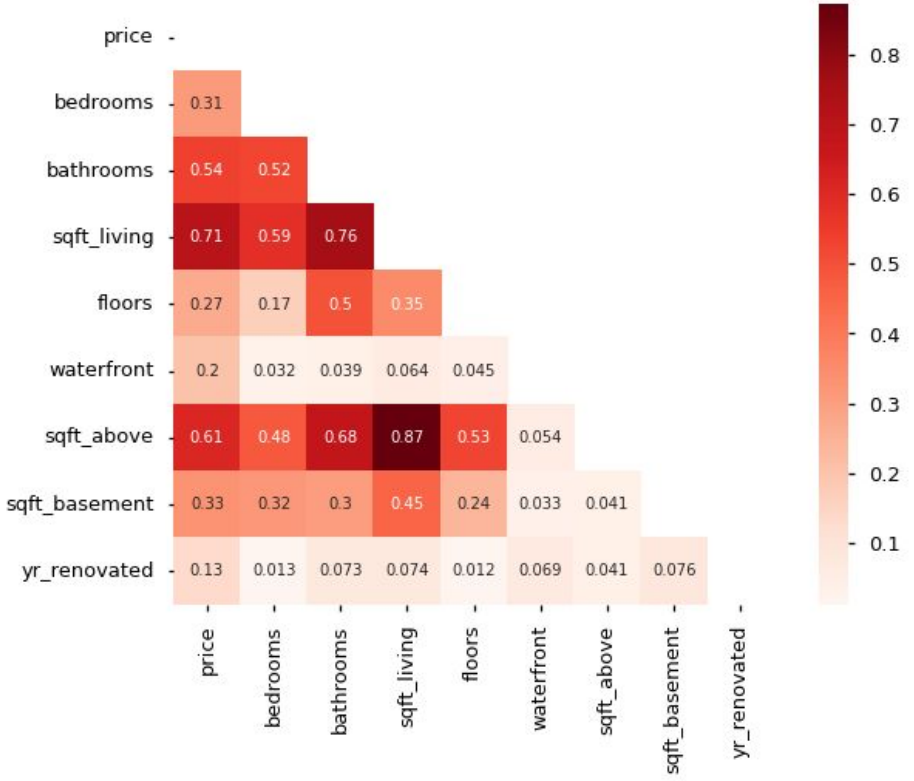
- When all else fails:
  - Try a different type of regression!

Independence of features

(AKA Little-to-No Multicollinearity)

That the features are not strongly related to other features.

- To Check:
  - Use correlation heatmaps!
  - Look for condition number warning on model.summary()



If your features violate no multicollinearity

- Change argument for your OneHotEncoder:
  - Add drop='first"
  - But can't use handle_unknown='ignore'

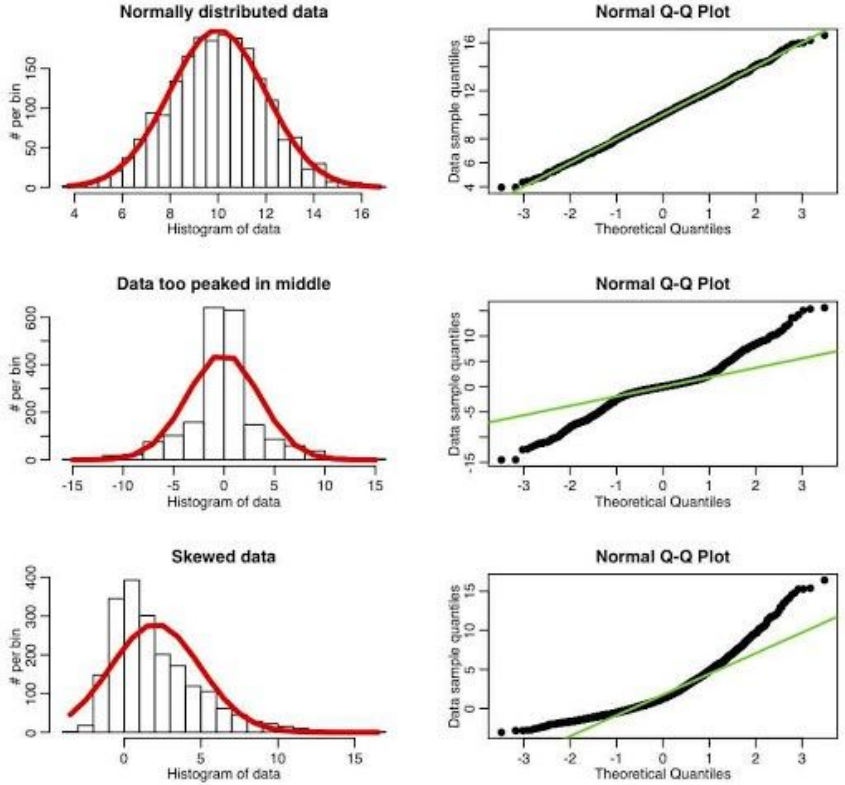
- Use Variance Inflation Factor! (will demonstrate today)
  - Get VIF Values for your features.
    - Examples:
      - https://www.geeksforgeeks.org/detecting-multicollinearity-with-vif-python/
      - Drop features with a score > ~5

- Try creating "interaction" features to combine 2 features that are highly correlated. Chris Albon Article on Interactions

Normality (of residuals)

The model's residuals are approximately normally distributed.

- To Check:
  - Use a Quantile-Quantile (Q-Q) Plot!
- Resource:
  - https://towardsdatascience.com/q-q-plots-explained-5aa8495426c0

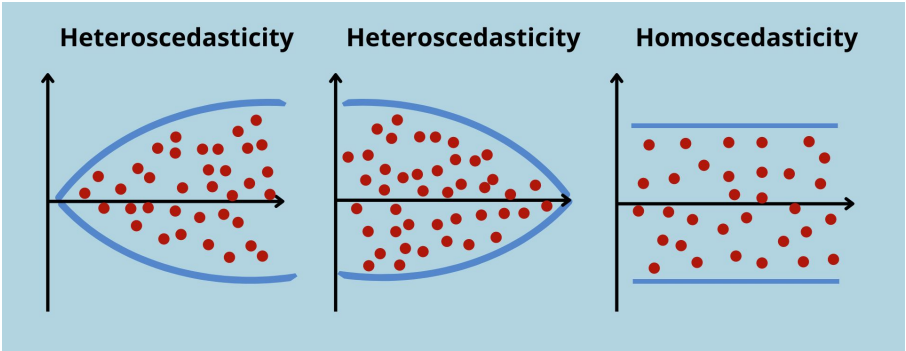


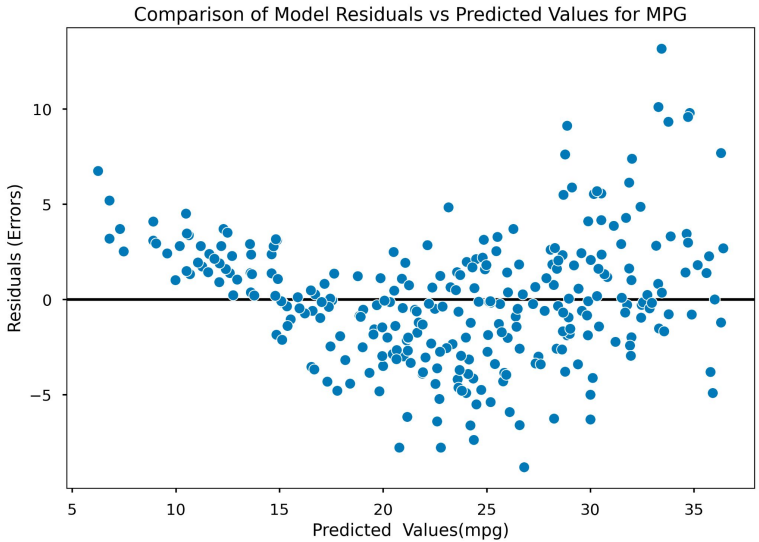
Source: Sherrytowers Q-Q plot examples

Homoscedasticity (Equal Variance)

The model residuals have equal variance across all predictions.

- To Check:
  - Plot a residual scatter plot!
    - X-axis: Predicted Y-Values
    - Y-axis: Residuals (y-y_pred)





Model Diagnostics: After Each OLS Model

- **Display the model.summary()**
  - Check R-Squared/Adj R-Squared
- **Get predictions and calculate residuals.**
- **Plot a QQ-Plot of the residuals to check for assumption of normally distributed residuals.**
- **Plot a scatter plot of the model's predictions (X) vs the residuals (y) to check for homoscedasticity.**
- **Check the P-Values of your coefficients.**
  - Drop any that are not significant.
  - OHE Columns are dropped All-or-None.
    - If the majority of the column's coefficient are significant, keep them all.
    - If the majority are not significant, drop them ALL.

If you violate either residual assumptions -1

Check for outliers in your target and your numeric features.

- 1) Try the Z-score rule for outliers first.

- 2) If still violating residual assumptions, try using the IQR outlier rule
  - We will demonstrate in activity
  - Perform on the ORIGINAL data, not the z-score outlier-removed data.

- 3) Revisit your numeric features and check if you have a nonlinear feature (if so, remove it).

  - Note: Remember to tell your stakeholders what range of your target was used in the model.
    - “My model was designed to predict homes with a value < $2million”

If you violate either residual assumptions-2

More extreme solutions:

- Forcing your numeric features to be more normally distributed using log-transformation..
  - Convert raw column to a log-transformed column (np.log())
  - Warning: changes meaning of coefficients from the effect of increasing the value of a feature by 1 to:
    - Then effect of a change of 1 PERCENT of a log-transformed feature.

- Use a special transformer from sklearn:
  - Quantile Transformer: Documentation
  - Visual Comparison of Transformers
  - Warning: will also change the interpretation of that features coefficients and I have no idea how to explain it to non-technical stakeholder!

Remember...

- We like linear regression for its simplicity and interpretability!
- I recommend:
  - using linear regression for explanations moreso than predictions.
  - avoiding transforming your features as much as possible
  - This includes avoiding scaling numeric features
    - (unless trying to use regularization like LassoRegression)

- ***"Perfect is the enemy of good" -*** *Voltaire*
  - We want a better QQ-Plot, not a perfectly flat one.
  - We want a no obvious cone shape to our residual plots, but a blob is ok!

End of Final Official Lecture

Image Source

Final Thoughts/Advice/Reminders

- **Attend tomorrow's bonus lecture, if you can!**

Discord:

- Send friend requests on Discord to your instructors and cohort mates.
  - Easier to stay in touch!

- Make sure to save any links or conversations you want to keep from the discord channel!

Post-Graduation

- 🤩 Check Out the Post-Graduation Resources!!!
  - Save the cohort notes repo: https://github.com/sensei-jirving/Online-DS-PT-01.24.22-cohort-notes
  - Save our YouTube Playlist: https://youtube.com/playlist?list=PLmeeqPbYmMC0XlmuN4agv0zvuAXP8HZS_

- DON'T STOP CODING AND DOING PROJECTS!
  - Re-do your first project from scratch in a new repo!

- Continue to practice SQL for job interviews!

Post-Class Activity Notebooks

- Class Notebook
- Solution Notebook (with very different final model)