

Sports Data Analysis

Week 4





TABLE OF CONTENTS

01

Icebreaker

02

Modeling

03

Model Types





Icebreaker

What is your favorite candy?

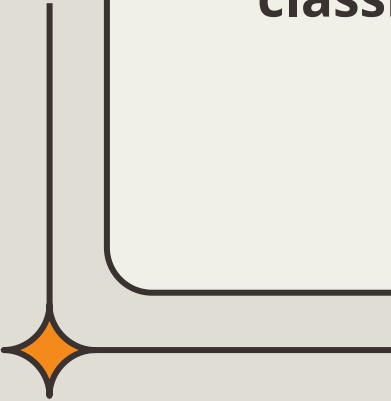
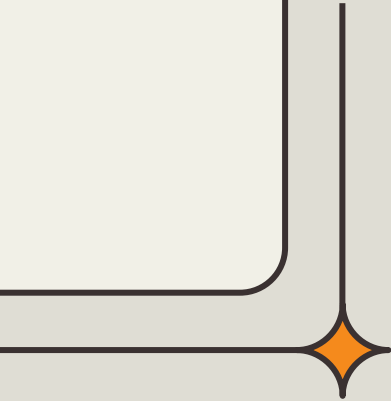
AND/OR

What is your comfort TV show?



Modeling

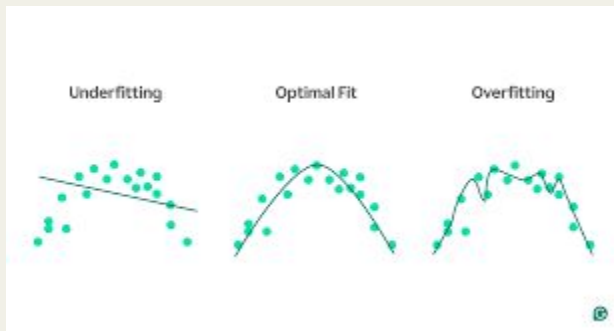
Modeling is (arguably) the most fun part of analytics

- Make **continuous** predictions (i.e. passing yards in a game, wins in a season) → **regression**
 - Group data into **classes** (i.e. win/loss, draft round) → **classification**
- 
- 

Overfitting

Overfitting happens when the model starts fitting to the **random noise** in the training data instead of just the relationships



- Leads to a lack of **generalization** outside of the train data







Preprocessing

Before building your model, you often have to “preprocess” the data to get it in a way that’s useful

- For categorical data, try **one-hot encoding** if the categories have no order and **label encoding** if they do (turns them into numbers)
 - **Scale/normalize** numerical features so big values don’t dominate little ones (passing yards doesn’t dominate interceptions thrown)
- 
- 

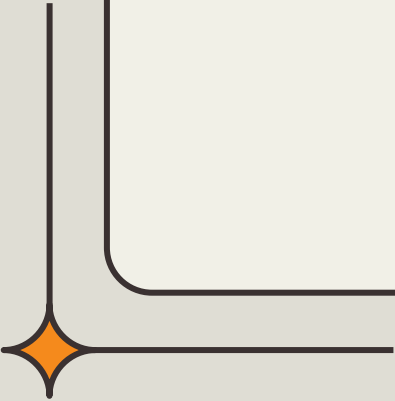
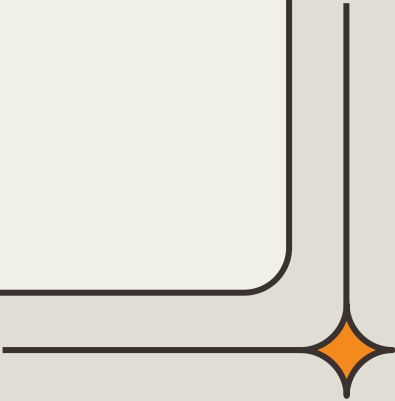


Preprocessing cntd.

- Scaling types include: **standardization** (z-scores), **min-max normalization** (puts the data between 0-1), and **robust scaling** (better if dealing with outliers)
 - With a lot of columns, try **dimensionality reduction** through **Principal Component Analysis** (fewest number of variables to explain most of the variance) or **Feature Selection** (chooses a subset of columns that contribute most to model performance)
- 
- 



Useful Functions

- **Encoding:** `get_dummies(df, columns=[])` for one-hot encoding, `LabelEncoder()` for label encoding
 - **Scaling:** `StandardScaler()`, `MinMaxScaler()`, `RobustScaler()`
 - **Dimensionality Reduction:** `PCA(n_components =)`, `SelectKBest(score_func = , k =)`
- 
- 

Types of Models

Linear Regression

- The regression that you are most used to!
- Works by trying to minimize the sum of squared residuals (may **overfit** to do this)

Lasso/Ridge Regression

- Uses a **penalization** term to try and mitigate overfitting
- Lasso regression performs **variable selection** by setting some slopes to 0
- Ridge **keeps all variables** and just minimizes slopes

Types of Models

K-nearest neighbors



- Classifies data points by using the classification of the **“k” nearest data points** from training and predicting the majority class

Logistic Regression

- Builds a linear model to predict the **probability of something happening**
- It then chooses which class is most likely using that and can work for more than 2 classes



Some Useful Functions

- `train_test_split(X, y, test_size)` → as the name suggests, splits your data into a training subset and a testing subset (helpful to **avoid overfitting**)
 - `predict_proba(X)` → gives you the **probabilities** associated with your classification with new data and a trained model
 - `LinearRegression()`, `LogisticRegression()`, `Ridge()`, `Lasso()`, `KNeighborsClassifier()`
- 
- 

Questions?



CREDITS: This presentation template was created by [Slidesgo](#), and includes icons by [Flaticon](#), and infographics & images by [Freepik](#)



Please keep this slide for attribution





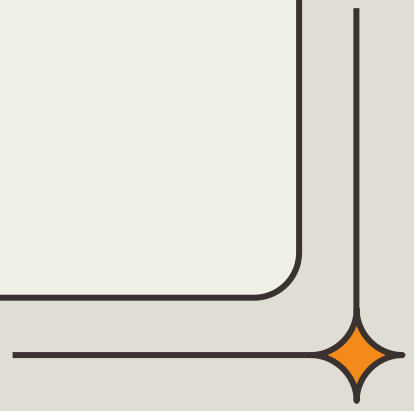
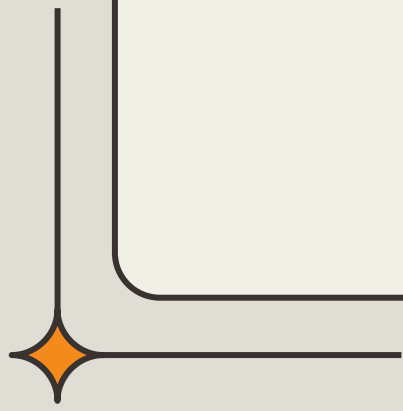
Goals for This Week

We're going to begin the model building process!

1. If you haven't yet, make a **correlation matrix** and **correlation heatmap** (look at last week's example code!)
 2. **Preprocess** your data (encode, standardize, maybe reduce dimensionality) for columns that **correlate to your chosen response variable** (look at this week's example code!)
 3. If you have time, start making a model (can use the starter code or keep going on your current file)
- 
- 



Example Code





Starter Code

<https://github.com/MichiganDataScienceTeam/F25-SportsDataAnalysis/blob/main/Starter%20Code/modeling.ipynb>

