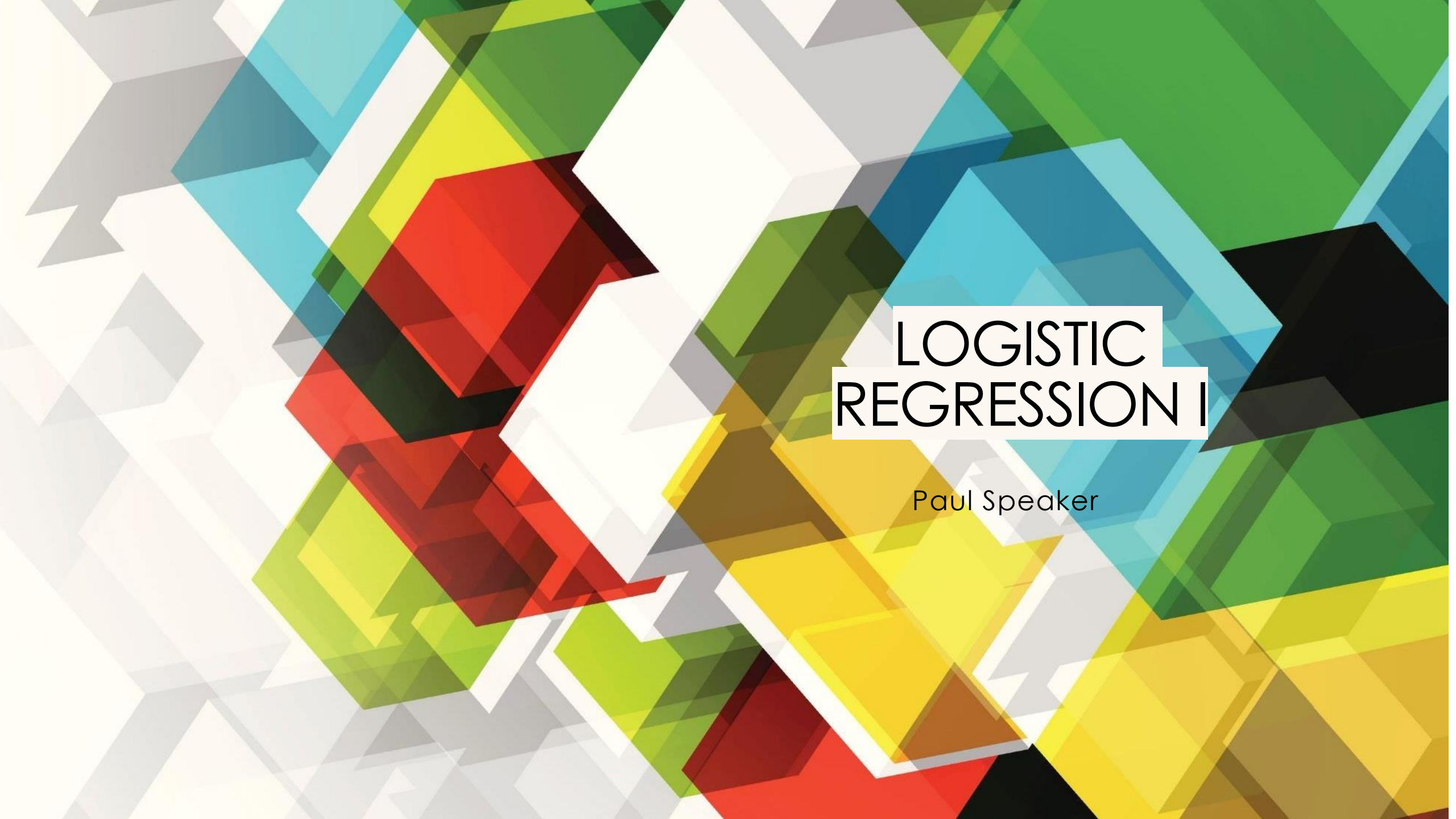
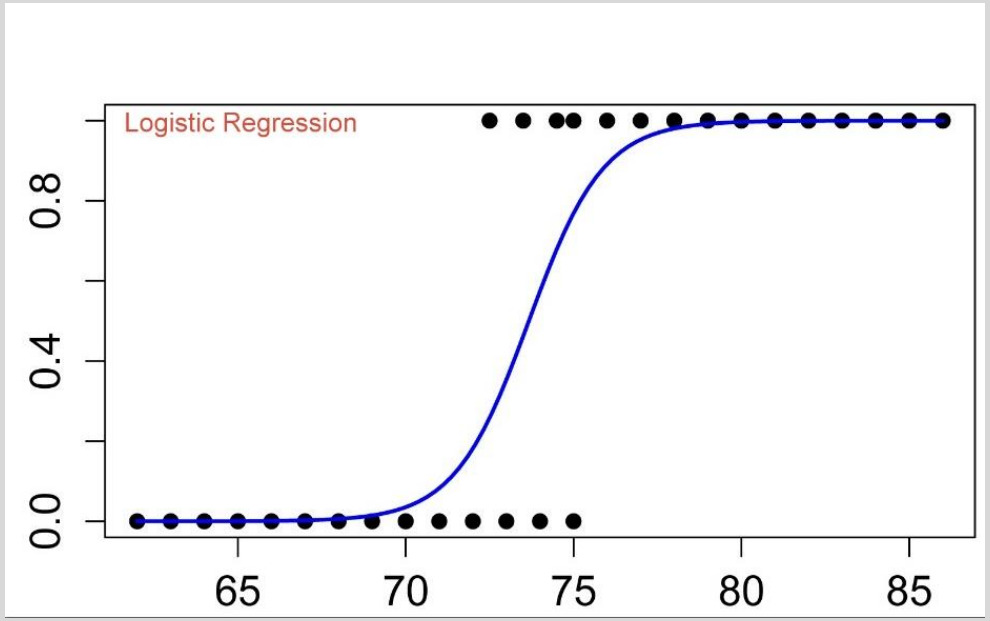

LOGISTIC
REGRESSION I
Paul Speaker

Logistic Regression

- Logistic regression is a binary classification model that uses numerical and categorical inputs to model the probability of being in one state or the other
- Based on the "log odds"
- Logistic transform to convert problem to continuous function

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

- Still intrinsically nonlinear, so optimization has to be done a little differently
- Monotonic, so still unique solutions



Logistic Regression & Conditional Probability

- The "prediction" of logistic regression is a probability of being in a certain class
- This probability is based on knowledge of x's, so it is not an unconditional probability
- Probability(Y = class 1 given that x = <the x values>)
  - Written as P(Y = 1 | X = x)
- Different knowledge of x's → different probability
  - Fundamentally Bayesian
- Other models look at the reverse probability P(X = x | Y = 1) and use Bayes rule to flip it around
  - More on this next week!
- Often (especially in medical field) the probability is expressed as an odds ratio (similar to odds given is sports betting): ratio = p/(1 – p) = P(yes)/P(no)

Logistic Regression Results

- Output of a logistic regression model is a probability
- Conventional way to use is to convert probability to binary target
  - $p > 0.5$ → one level
  - $p < 0.5$ → other level
  - Might use different cutoff if cost of false positive different than for false negative
- Probabilities are also useful directly
  - Suppose you are identifying target customers.
    - If you can only target some of the yes in model, might be better to target those with high probability values
  - Suppose you are predicting sales for 5 customers. Model give a 0.4 probability of sale for each.
  - What is better prediction: 0 sales (all predicted not to be sales) or 2 sales (n * p)?
  - What to use depends on whether you are looking at aggregate or individual behavior

Logistic Regression in R

- Logistic regression is done with glm command (no library required)
- Format: glm(<model form>, data = <>, family = “binomial”)
- Outputs are very similar for lm
- Predictions and residuals are in 0-1 range
- Predict command can also be used (especially with test dataset)
  - One difference: form is predict(<model dataframe> , <new data>, type = “response”)
    - Needed for stupid reasons (should be default)
- Other useful way to look at output: confusion matrix

Logistic Regression in R

- glm allows logistic regression models (and several others)
  - No library needed
- Syntax:
  - glm(data = <dataframe>, <target> ~ x1 + <x2> + …, family = binomial)
  - family = binomial → identifies it as a logistic regression model
    - Number of successes as function of the x's
  - First 2 arguments are identical to linear regression in R
  - The y target should be numeric with values of 0 and 1.
    - Conversion is often needed
- Like linear regression, there are a lot of different outputs
  - Parameter estimates
  - Fitted values (probability of being in y = 1 class)

Logistic Regression Predictions

- Given a logistic regression model, there are two ways to make predictions off of new data
  - Directly, using the formula
  - The predict function in R
- Directly: Recall that the form of logistic regression is

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

  - β_0 is the intercept, β_1 is the coefficient of x. You put in the x value(s), the output is the actual probability
- Syntax for predict function:
  - predict(<model object>, <dataframe with x's>, <type = 'response'>)
    - If type = 'response' is not set, output will be in terms of odds ratio, not probability

Confidence Intervals

- The summary output of a logistic regression model can look like this:

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.839597   1.498563   3.229  0.00124 **
Age          0.019706   0.015339   1.285  0.19889
MaxHR       -0.040711   0.006808  -5.980 2.24e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 417.98  on 302  degrees of freedom
Residual deviance: 359.24  on 300  degrees of freedom
AIC: 365.24
```

- Notice the “z value” column (not “t value” like in linear regression)
  - Since the output is a proportion, the confidence intervals are built with normal distribution, not t distribution
  - CI (Age) = 0.019706 + 0.015339*qnorm(c(0.025, 0.975)) = qnorm(c(0.025, 0.975), 0.019706, 0.015339)

Confusion Matrix

- The confusion matrix gives a view to how well the predictions work for any possible target classes
- We will use the caret package to produce confusion matrices
- Some work to get data in right form
  - Both actual and predicted have to be in factor format
  - Predicted is probability → has to be converted to actual class prediction
  - as.factor(as.integer(2*<predictions>))
- ConfusionMatrix command produces a lot of outputs besides confusion matrix
  - Accuracy(ratio of true to false overall)
  - Other ratios are conditional probabilities like
    - P(A | B)
    - P(B | A)