```
  Course: Regression_Models
  Lesson: Binary_Outcomes


- Class: text
  Output: "Binary Outcomes. (Slides for this and other Data Science courses may be
  found at github https://github.com/DataScienceSpecialization/courses. If you care to
  use them, they must be downloaded as a zip file and viewed locally. This lesson
  corresponds to Regression_Models/03_02_binaryOutcomes.)"

- Class: text
  Output: Frequently we care about outcomes that have two values such as alive or dead,
  win or lose, success or failure. Such outcomes are called binary, Bernoulli, or 0/1.
  A collection of exchangeable binary outcomes for the same covariate data are called
  binomial outcomes. (Outcomes are exchangeable if their order doesn't matter.)

- Class: text
  Output: In this unit we will use glm() to model a process with a binary outcome and a
  continuous predictor. We will also learn how to interpret glm coefficients, and how
  to find confidence intervals. But first, let's discuss odds.

- Class: text
  Output: The Baltimore Ravens are a team in the American Football League. In post
  season (championship) play they win about 2/3 of their games. In other words, they
  win about twice as often as they lose. If I wanted to bet on them, I would have to
  offer 2-to-1 odds--if they lost I would pay you $2, but if they won you would pay me
  only $1. That way, in the long run over many bets, we'd both expect to win as much
  money as we'd lost.

- Class: mult_question
  Output: During the regular season the Ravens win about 55% of their games. What odds
  would I have to offer in the regular season?
  AnswerChoices: 55 to 45; 11 to 9; 1.22222 to 1; Any of these
  CorrectAnswer: Any of these
  AnswerTests: omnitest()
  Hint: Any answer will do.

- Class: text
  Output: All of the answers are correct because they all represent the same ratio. If
  p is the probability of an event, the associated odds are p/(1-p).

- Class: figure
  Output: Now suppose we want to see how the Ravens' odds depends on their offense. In
  other words, we want to model how p, or some function of it, depends on how many
  points the Ravens are able to score. Of course, we can't observe p, we can only
  observe wins, losses, and the associated scores. Here is a Box plot of one season's
  worth of such observations.
  Figure: nevermore.R
  FigureType: new

- Class: mult_question
  Output: We can see that the Ravens tend to win more when they score more points. In
  fact, about 3/4 of their losses are at or below a certain score and about 3/4 of
  their wins are at or above it. What score am I talking about? (Remember that the
  purple boxes represent 50% of the samples, and the "T's" 25%.)
  AnswerChoices: 23;18;30;40
  CorrectAnswer: 23
  AnswerTests: omnitest(correctVal='23')
  Hint: The purple "loss" box is to the left of this score and the purple "win" box to
  its right.

- Class: figure
  Output: There were 9 games in which the Ravens scored 23 points or less. They won 4
  of these games, so we might guess their probability of winning, given that they score
  23 points or less, is about 1/2.
  Figure: purple_line.R
  FigureType: add

- Class: cmd_question
```

```
    Output: There were 11 games in which the Ravens scored 24 points or more. They won
    all but one of these. Verify this by checking the data yourself. It is in a data
    frame called ravenData. Look at it by typing either ravenData or View(ravenData).
    CorrectAnswer: ravenData
    AnswerTests: ANY_of_exprs('ravenData', 'View(ravenData)')
    Hint: Type ravenData to print the data in the console. Type View(ravenData) to see it
    a separate window.

- Class: figure
    Output: We see a fairly rapid transition in the Ravens' win/loss record between 23
    and 28 points. At 23 points and below they win about half their games, between 24 and
    28 points they win 3 of 4, and above 28 points they win them all. From this, we get a
    very crude idea of the correspondence between points scored and the probability of a
    win. We get an S shaped curve, a graffiti S anyway.
    Figure: graffiti_s.R
    FigureType: new

- Class: text
    Output: Of course, we would expect a real curve to be smoother. We would not, for
    instance, expect the Ravens to win half the games in which they scored zero points,
    nor to win all the games in which they scored more than 28. A generalized linear
    model which has these properties supposes that the log odds of a win depend linearly
    on the score. That is, log(p/(1-p)) = b0 + b1*score. The link function, log(p/(1-p)),
    is called the logit, and the process of finding the best b0 and b1, is called
    logistic regression.

- Class: text
    Output: 'The "best" b0 and b1 are those which maximize the likelihood of the actual
    win/loss record. Based on the score of a game, b0 and b1 give us a log odds, which we
    can convert to a probability, p, of a win. We would like p to be high for the scores
    of winning games, and low for the scores of losses.'

- Class: cmd_question
    Output: We can use R's glm() function to find the b0 and b1 which maximize the
    likelihood of our observations. Referring back to the data frame, we want to predict
    the binary outcomes, ravenWinNum, from the points scored, ravenScore. This
    corresponds to the formula, ravenWinNum ~ ravenScore, which is the first argument to
    glm. The second argument, family, describes the outcomes, which in our case are
    binomial. The third argument is the data, ravenData. Call glm with these parameters
    and store the result in a variable named mdl.
    CorrectAnswer: 'mdl <- glm(ravenWinNum ~ ravenScore, binomial, ravenData)'
    AnswerTests: creates_glm_model('mdl <- glm(ravenWinNum ~ ravenScore, binomial,
    ravenData)')
    Hint: Use an expression such as mdl <- glm(ravenWinNum ~ ravenScore, binomial,
    ravenData) or mdl <- glm(ravenWinNum ~ ravenScore, family=binomial, data=ravenData).

- Class: figure
    Output: "The probabilities estimated by logistic regression using glm() are
    represented by the black curve. It is more reasonable than our crude estimate in
    several respects: It increases smoothly with score, it estimates that 15 points give
    the Ravens a 50% chance of winning, that 28 points give them an 80% chance, and that
    55 points make a win very likely (98%) but not absolutely certain."
    Figure: glm_vs_graffiti.R
    FigureType: new

- Class: cmd_question
    Output: "The model is less credible at scores lower than 9. Of course, there is no
    data in that region; the Ravens scored at least 9 points in every game. The model
    gives them a 33% chance of winning if they score 9 points, which may be reasonable,
    but it also gives them a 16% chance of winning even if they score no points! We can
    use R's predict() function to see the model's estimates for lower scores. The
    function will take mdl and a data frame of scores as arguments and will return log
    odds for the give scores. Call predict(mdl, data.frame(ravenScore=c(0, 3, 6))) and
    store the result in a variable called lodds."
    CorrectAnswer: 'lodds <- predict(mdl, data.frame(ravenScore=c(0, 3, 6)))'
    AnswerTests: expr_creates_var('lodds');omnitest('lodds <- predict(mdl,
    data.frame(ravenScore=c(0, 3, 6)))')
    Hint: Type lodds <- predict(mdl, data.frame(ravenScore=c(0, 3, 6))) to produce the
    model's estimated log odds of a win for scores 0, 3, and 6.
```

```
- Class: cmd_question
  Output: "Since predict() gives us log odds, we will have to convert to probabilities. To convert log odds to probabilities use exp(lodds)/(1+exp(lodds)). Don't bother to store the result in a variable. We won't need it."
  CorrectAnswer: 'exp(lodds)/(1+exp(lodds))'
  AnswerTests: omnitest('exp(lodds)/(1+exp(lodds))')
  Hint: Type exp(lodds)/(1+exp(lodds)) to convert the log odds, lodds, to probabilities. This expression is called the inverse logit function.

- Class: cmd_question
  Output: "As you can see, a person could make a lot of money betting against this model. When the Ravens score no points, the model might like 16 to 84 odds. As it turns out, though, the model is not that sure of itself. Typing summary(mdl) you can see the estimated coefficients are both within 2 standard errors of zero. Check out the summary now."
  CorrectAnswer: summary(mdl)
  AnswerTests: omnitest('summary(mdl)')
  Hint: Just type summary(mdl).

- Class: text
  Output: "The coefficients estimate log odds as a linear function of points scored. They have a natural interpretation in terms of odds because, if b0 + b1*score estimates log odds, then exp(b0 + b1*score)=exp(b0)exp(b1*score) estimates odds. Thus exp(b0) is the odds of winning with a score of 0 (in our case 16/84,) and exp(b1) is the factor by which the odds of winning increase with every point scored. In our case exp(b1) = exp(0.10658) = 1.11. In other words, the odds of winning increase by 11% for each point scored."

- Class: cmd_question
  Output: "However, the coefficients have relatively large standard errors. A 95% confidence interval is roughly 2 standard errors either side of a coefficient. R's function confint() will find the exact lower and upper bounds to the 95% confidence intervals for the coefficients b0 and b1. To get the corresponding intervals for exp(b0) and exp(b1) we would just exponentiate the output of confint(mdl). Do this now."
  CorrectAnswer: 'exp(confint(mdl))'
  AnswerTests: omnitest('exp(confint(mdl))')
  Hint: Just type exp(confint(mdl)).

- Class: mult_question
  Output: "What is the 2.5% confidence bound on the odds of winning with a score of 0 points?"
  AnswerChoices: 0.005674966;0.996229662;2.5%
  CorrectAnswer: '0.005674966'
  AnswerTests: omnitest(correctVal= '0.005674966')
  Hint: It's very small.

- Class: mult_question
  Output: "The lower confidence bound on the odds of winning with a score of 0 is near zero, which seems much more realistic than the 16/84 figure of the maximum likelihood model. Now look at the lower bound on exp(b1), the exponentiated coefficient of ravenScore. How does it suggest the odds of winning will be affected by each additional point scored?"
  AnswerChoices: They will decrease slightly;They will increase slightly;They will increase by 30%
  CorrectAnswer: They will decrease slightly
  AnswerTests: omnitest(correctVal= 'They will decrease slightly')
  Hint: If you multiply a positive number by 0.996229662, do you increase or decrease the value?

- Class: text
  Output: "The lower confidence bound on exp(b1) suggests that the odds of winning would decrease slightly with every additional point scored. This is obviously unrealistic. Of course, confidence intervals are based on large sample assumptions and our sample consists of only 20 games. In fact, the GLM version of analysis of variance will show that if we ignore scores altogether, we don't do much worse."

- Class: cmd_question
```

```
    Output: "Linear regression minimizes the squared difference between predicted and
    actual observations, i.e., minimizes the variance of the residual. If an additional
    predictor significantly reduces the residual's variance, the predictor is deemed
    important. Deviance extends this idea to generalized linear regression, using
    (negative) log likelihoods in place of variance. (For a detailed explanation, see the
    slides and lectures.) To see the analysis of deviance for our model, type anova(mdl)."
    CorrectAnswer: 'anova(mdl)'
    AnswerTests: omnitest('anova(mdl)')
    Hint: Type anova(mdl).

- Class: cmd_question
    Output: "The value, 3.5398, labeled as the deviance of ravenScore, is actually the
    difference between the deviance of our model, which includes a slope, and that of a
    model which includes only an intercept, b0. This value is centrally chi-square
    distributed (for large samples) with 1 degree of freedom (2 parameters minus 1
    parameter, or equivalently 19-18.) The null hypothesis is that the coefficient of
    ravenScore is zero. To confidently reject this hypothesis, we would want 3.5398 to be
    larger than the 95th percentile of chi-square distribution with one degree of
    freedom. Use qchisq(0.95, 1) to compute the threshold of this percentile."
    CorrectAnswer: 'qchisq(0.95, 1)'
    AnswerTests: ANY_of_exprs('qchisq(0.95, 1)', 'qchisq(.95, 1)')
    Hint: Type qchisq(0.95, 1).

- Class: text
    Output: "As you can see, 3.5398 is close to but less than the 95th percentile
    threshold, 3.841459, hence would be regarded as consistent with the null hypothesis
    at the conventional 5% level. In other words, ravenScore adds very little to a model
    which just guesses that the Ravens win with probability 70% (their actual record that
    season) or odds 7 to 3 is almost as good. If you like, you can verify this using mdl0
    <- glm(ravenWinNum ~ 1, binomial, ravenData), but this concludes the Binary Outcomes
    example. Thank you."

- Class: mult_question
    Output: "Would you like to receive credit for completing this course on
      Coursera.org?"
    CorrectAnswer: NULL
    AnswerChoices: Yes;No
    AnswerTests: coursera_on_demand()
    Hint: ""
```