SOC-GA 2332 Intro to Stats Lab 9

Risa Gelles-Watnick

10/30/2025

Part 0: Logistics

- Due date for **problem set #3** moved to Nov. 21st (11:59pm)
- Lab feedback survey

Agenda

- Review quiz #3
- Discrete dependent variables
- Probability linear model
- Logistic regression

Part 2: Discrete Dependent Variable

```
knitr::opts_chunk$set(echo = TRUE,
                      cache = FALSE,
                      fig.align = "center",
                      fig.width = 4.5,
                      fig.height = 4,
                      letina = TRUE)

pacman::p_load(
  tidyverse,
  janitor,
  kableExtra # for tables
)
```

2.1 Binary Response

- Let's focus on the case when the dependent variable is a binary response (0 and 1). For example, we may be interested in whether one voted for Trump in the 2024 General Election (`trump == 1` if Yes)
- As in regular OLS, we may also be interested in the association between income, gender, education, and party identification (strong Democratic = 0) on whether R voted for Trump.
  - `pid` is party ID, with 0 being a strong Democrat and 6 being a strong Republican

```
## load data
load("data/dat.RData")
```

Exercise 1 (10 minutes)

What statistics/information might be helpful to have before we analyze this data?

- what each variable means
- number of observations
- number of observations in certain categories of interest
- how explanatory variables correlate to outcome variable
- which variables have missingness and how much

There are many things we might want to explore about this data before we run analyses. Pick one preliminary exploration (ex. summary statistics, missing data, etc.) and create a visualization (either a plot or a clean table). **Hint 1**: the `summarize()` & `group_by()` commands are often helpful for summary stats. **Hint 2**: you might want to use the `kableExtra` package for clean tables. For example:

```
# creating fake dataframe that we want to put in a table
fake_data <- data.frame("Variabble A" = c(1,2), "Variable B" = c("Yes", "No"))

# clean table with kableExtra
fake_data %>%
    kable(digits = 4, # number of digits to display
          caption = "Fake Data Table") %>% # title for table
    kable_styling(
      bootstrap_options = "striped", # add grey stripes to alternate rows
      full_width = FALSE             # allow table to be narrower than full page
      )
```

Table 1: Fake Data Table

Variabble.A	Variable.B
1	Yes
2	No

```
# creating summary stats
sum_stats <- dat %>%
  group_by(pid) %>%
  summarize(
    "N" = n(),
    "% Voted Trump" = mean(trump, na.rm = TRUE)
  )

# plotting summary stats
sum_stats %>%
    kable(digits = 4, # number of digits to display
          caption = "Election Data Summary Statistics") %>% # title for table
    kable_styling(
      bootstrap_options = "striped", # add grey stripes to alternate rows
      full_width = FALSE             # allow table to be narrower than full page
      )
```

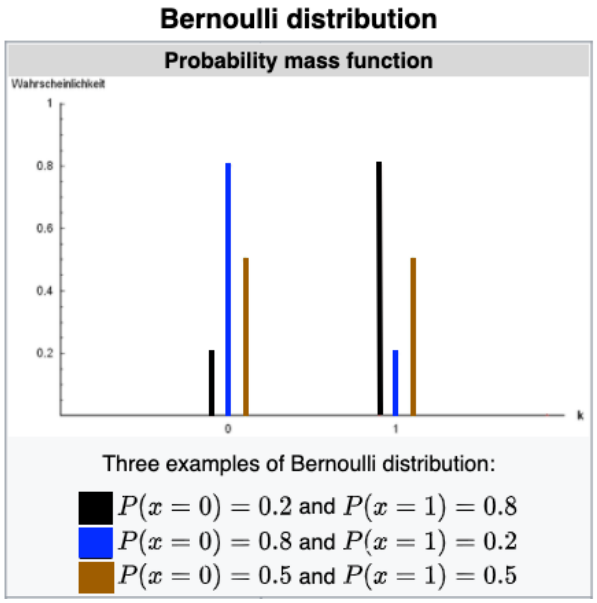
Table 2: Election Data Summary Statistics

pid	N	% Voted Trump
0	379	0.0288
1	352	0.1152
2	307	0.1069
3	414	0.3900
4	259	0.8640

5	272	0.8539
6	255	0.9764
NA	110	0.5405

2.2 Bernoulli Distribution

- When the outcome is binary, $Y_i \in \{0, 1\}$, it is common to assume that it follows a Bernoulli distribution. A Bernoulli distribution has one parameter, which we might call π, that represents the probability of a "success." It is the convention to let $Y_i = 1$ be a success and $Y_i = 0$ be a fail; so, $\pi_i = \Pr[Y_i = 1]$.
- An example of the Bernoulli distribution is the toss of a biased coin. We can plot the expected probability of whether we get a "tail" or "head" given how the coin is biased.



- To model binary outcomes, we are assuming the observed outcome follows a Bernoulli distribution for each unit of observation i with a parameter π that can be predicted with a set of predictors, $\boldsymbol{X}_i$.

2.3 The Linear Probability Model (LPM)

- Suppose that we have a binary outcome $Y_i \in \{0, 1\}$, where we think that the success probability depends on a set of predictors $\boldsymbol{X}_i = \{X_{i1}, X_{i2}, ..., X_{i3}\}$. We might write this as $\pi_i = \Pr[Y_i = 1|\boldsymbol{X}_i] = \pi(\boldsymbol{X}_i)$, where the subscript i shows that the success probability will vary across units in our sample, and where we have emphasized that $\pi(\cdot)$ is a function of a set of predictors.
- In the LPM, we assume that $\pi(\boldsymbol{X}_i)$ is a **linear** function of the predictors. That is,

$$\pi(\boldsymbol{X}_i) = \mathbb{E}[Y_i|\boldsymbol{X}_i] = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_k X_{ik}, \qquad i = 1, 2, ..., n.$$

- Notice that the equation looks *exactly* the same as the linear regression model we have considered in the previous labs. The only difference is that the outcome is a binary variable.
- Linear Probability Model can be estimated using the same method as we estimate a regular linear model. In `R`, simply use `lm()`. In our case, we model the outcome variable `trump` using all the predictors in the dataframe.

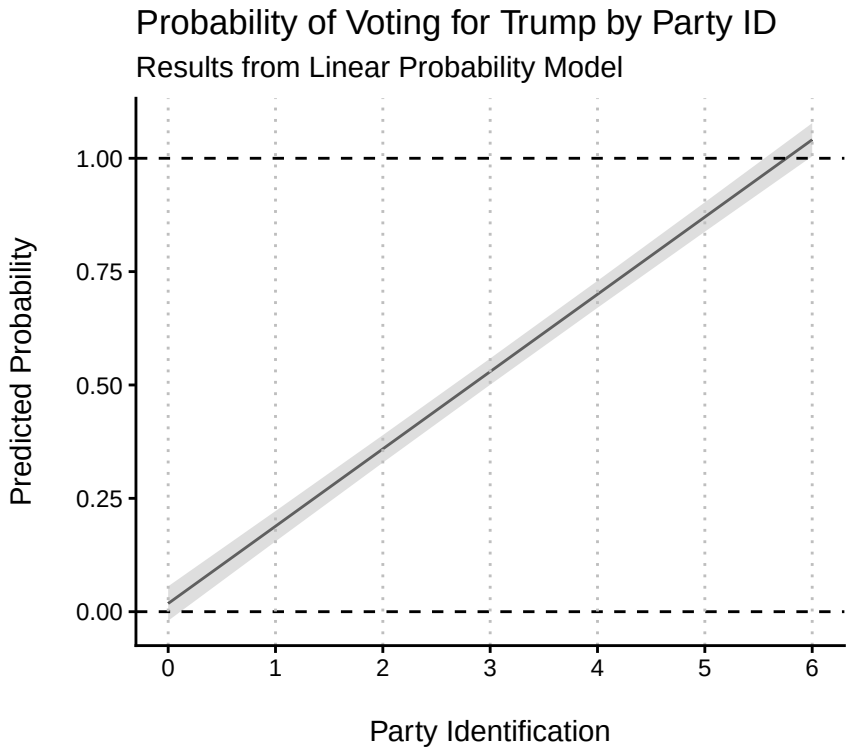
```r
## fit LPM
lpm <- lm(trump ~ pid + log_inc + female + college, data = dat)

## print summary
summary(lpm)
```

```
##
## Call:
## lm(formula = trump ~ pid + log_inc + female + college, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0409 -0.1428  0.0118  0.1296  1.0514
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.290119   0.097759  -2.968 0.003059 **
## pid          0.170488   0.003975  42.889  < 2e-16 ***
## log_inc      0.028229   0.009199   3.069 0.002198 **
## female      -0.042548   0.017531  -2.427 0.015369 *
## college     -0.066244   0.019320  -3.429 0.000626 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3036 on 1217 degrees of freedom
##   (1126 observations deleted due to missingness)
## Multiple R-squared:  0.6245, Adjusted R-squared:  0.6232
## F-statistic:   506 on 4 and 1217 DF,  p-value: < 2.2e-16
```

- How would we interpret the pid coefficient of this regression? The interpretation of the coefficients of LPM is straightforward: Holding other variables constant, one unit increase in X_k will increase/decrease the probability of $Y = 1$ by β_k. So one unit increase in pid is associated with a .17 increase in the probability of voting Trump on average.
- We can plot the LPM's predicted probability of voting for Trump by Party ID.

```r
## create new dataset for predictions
pred_dat <- data.frame(
  pid = 0:6,  # pid values to get predictions for
  log_inc = median(dat$log_inc, # fix inc at median
                   na.rm = T),
  female = 0, # fix gender at male
  college = 0 # fix education at less than BA
)

## create a new df of predict probability of voting for Trump
yhat <- cbind(pid = 0:6,
              predict(lpm, newdata = pred_dat, type = "response",
                      interval = "confidence")) %>%
  as_tibble()

## plot predicted probabilities (save plot in object lpm_plot)
lpm_plot <- yhat %>%
  ggplot(aes(x = pid, y = fit)) +
  geom_line(col = "black") +
```

```r
  geom_ribbon(aes(ymin = lwr, ymax = upr),
              fill = "grey", alpha = .5, col = NA) +
  scale_y_continuous(
    name = "Predicted Probability\n",
    breaks = seq(0, 1, .25)) +
  scale_x_continuous(
    name = "\nParty Identification",
    breaks = seq(0, 6, 1)) +
  geom_hline(yintercept = c(0, 1),linetype = 2) +
  geom_vline(xintercept = seq(0, 6, 1), linetype = 3, col = "grey") +
  theme_classic() +
  ggtitle("Probability of Voting for Trump by Party ID",
          subtitle = "Results from Linear Probability Model")

## print the plot
print(lpm_plot)
```

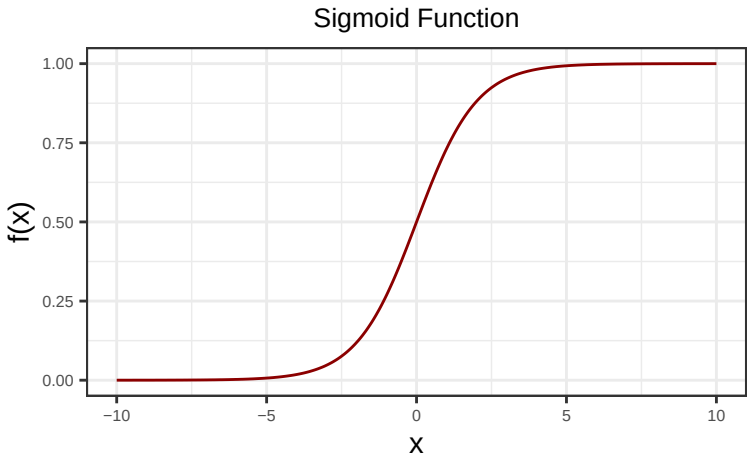


- There are three things to notice:
  - As the name of the model suggests, the predicted probability of voting for Trump is increasing *linearly* with our predictor.
  - We see that the confidence interval at `pid = 0` (i.e., "Strong Democrats") and both the confidence interval and our fitted value at `pid = 6` (i.e., "Strong Republican") take on *impossible values*. There cannot be something like a probability that is smaller than zero or larger than 1.
  - The error distribution will not be homoskedastic
- The above issues (especially the second one) motivate a **logistic regression model** (or a **Generalized Linear Model (GLM)**) in which the predicted probabilities are guaranteed to lie between zero and one, and we allow semi-*non-linear* relations between our predictors and the outcome.

Part 3: Logistic Regressions

3.1 Motivations and Basics

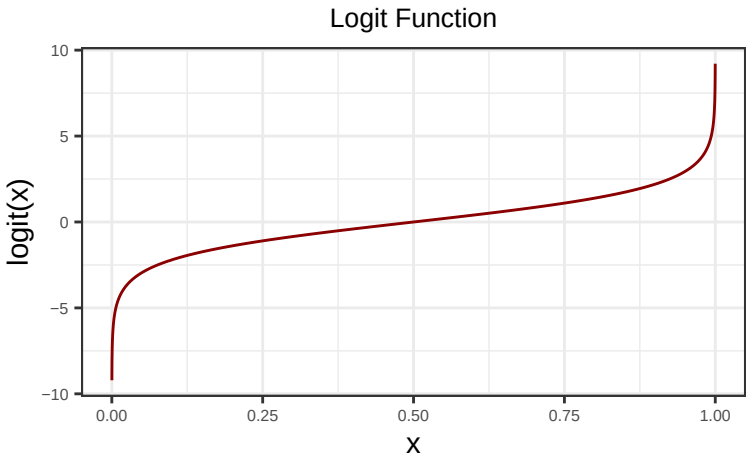
- To bound the predicted probability within 0 and 1, we turn to the Sigmoid Function
- $f(x) = \frac{1}{1+e^{-x}}$



- The Sigmoid Function has two desired properties
  - 1. $f(x)$ is bounded within 0 and 1
  - 2. x has no limit
- The Sigmoid Function is therefore a good candidate to model the **probabilities** of some categorical dependent variable (e.g., voted for Trump $\pi_i = \pi(\boldsymbol{X}_i)$), given some observed characteristics $\boldsymbol{X}_i$
- We specify the predicted conditional probability $\pi_i = \frac{1}{1+e^{-(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})}}$
- With some algebra
- $1-\pi_i = 1 - \frac{1}{1+e^{-(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})}} = \frac{e^{-(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})}}{1+e^{-(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})}}$
- $\frac{\pi_i}{1-\pi_i} = \frac{1}{e^{-(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})}}$
- $\text{logit}(\pi_i) = \log(\frac{\pi_i}{1-\pi_i}) = \log\left(\frac{1}{e^{-(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})}}\right) = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_k X_{ik}$
- This is the logit transformation!

3.2 Properties of Sigmoid and Logit

- Sigmoid function and logit function are inverse functions for each other
- Sigmoid function: $y = \frac{1}{1+e^{-x}}$
- Inverse of Sigmoid function: $x = \frac{1}{1+e^{-y}} \rightarrow y = \log(\frac{x}{1-x})$
- $\frac{1}{1+e^{-x}}$ is bounded within 0 and 1. Inversely, the x in $\log(\frac{x}{1-x})$ is bounded within 0 and 1



3.3 Odds and Odds Ratios

- We call the term $\frac{\pi_i}{1-\pi_i}$ in the log() function "odds" (probability of "event" divided by probability of no "event")
- Odds $Odds = \frac{\pi_i}{1-\pi_i} = \exp(\beta_0 + \beta_1 X_{i1} + \cdots + \beta_k X_{ik})$
- We introduce the notion of odds ratio to interpret β_k
- Odds ratio $= \frac{Odds_{X_{i1}+1}}{Odds_{X_{i1}}} = \frac{\exp(\beta_0+\beta_1(X_{i1}+1)+\cdots+\beta_k X_{ik})}{\exp(\beta_0+\beta_1 X_{i1}+\cdots+\beta_k X_{ik})} = \exp(\beta_1)$
  - You may find it analogous to OLS, where $\hat{\beta}_1$ describes the (additive) change of the dependent variable when the independent variable change by 1 unit

3.4 Estimate Logistic Regression in R

- Fitting a logistic regression in R is straightforward. If we use the same predictors as those of the LPM discussed above, the code to fit the logistic regression is.

```
## fit logistic regression model (glm)
l_reg <- glm(
    trump ~ pid + log_inc + female + college, # formula of regression
    family = binomial(link = "logit"),        # specifying the dist. of outcome
    data = dat                                # data
)

## check the class of the object
class(l_reg)
```

```
## [1] "glm" "lm"
```

```
## summarize results
summary(l_reg)
```

```
##
## Call:
## glm(formula = trump ~ pid + log_inc + female + college, family = binomial(link = "logit"),
##     data = dat)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.47611    1.11850  -5.790 7.04e-09 ***
## pid          1.20627    0.06211  19.420  < 2e-16 ***
## log_inc      0.29218    0.10210   2.862 0.004215 **
```

```
## female      -0.46170    0.19580  -2.358 0.018372 *
## college     -0.76016    0.21755  -3.494 0.000476 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1666.84  on 1221  degrees of freedom
## Residual deviance:  709.81  on 1217  degrees of freedom
##   (1126 observations deleted due to missingness)
## AIC: 719.81
##
## Number of Fisher Scoring iterations: 6
```

3.5 Model Interpretation

- In the `summary` for logistic regression models:
  - 1. The coefficients in the `Estimate` column show the estimated regression coefficients, i.e., the $\hat{\beta}_k$'s. For example, the coefficient of the `pid` variable suggests that a unit increase in `pid` is associated with a `1.206` increase in the *logit* (the *log odds*) of the probability of voting for Trump.
  - 2. We might also interpret the coefficients in terms of *odds* by exponentiating them. Hence, the odds of the outcome are predicted to increase **by a factor of** $e^{\hat{\beta}_1}$ for each unit increase in `pid`. For example, a unit increase in `pid` is associated with an increase of the odds of voting for Trump by a factor of $e^{1.206} = 3.340$.
  - 3. You can calculating the exponentiated regression coefficients of a logistic regression model by using the `coef` function to extract the estimated coefficients from the model and, then, use the `exp()` function:

```r
## extract coefficients
l_coef = coef(l_reg)

## print coefficients and their exponentiated form
rbind(coef = l_coef, exp_coef = exp(l_coef))
```

```
##           (Intercept)      pid   log_inc     female    college
## coef     -6.476108455 1.206265 0.2921768 -0.4616960 -0.7601581
## exp_coef  0.001539791 3.340984 1.3393398  0.6302139  0.4675925
```

3.6 Predicted Probabilities

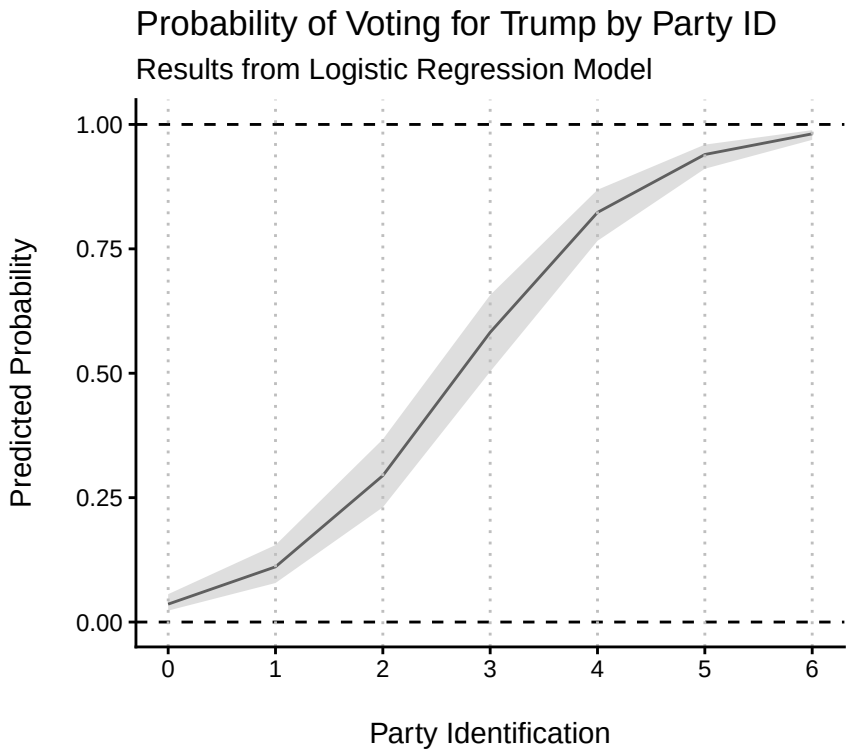
- It is always a good idea to plot the predicted probabilities (both for yourself and for your readers). In other words, we want to plot how the probability of the outcome changes when we vary a focal variable while fixing the remain variables at certain values.
- Conceptually, what is the difference between predicted probabilities versus the coefficients from our logistic regression? Predicted probabilities are the probabilities of "success" (outcome variable = 1) for specific groups. Ex. what is the probability that you vote trump if you're a strong democrat who is female and college educated. The coefficients from the logistic regression are a measure of how much each of these categories (strong democrat, female, college educated) change your probability of success (voting Trump).
- In R, doing this is quite straightforward. We have already created a new dataset for which we want the predictions above (when plotting the predicted probabilities using the LPM). Let us use the exact same dataset again.

```r
## predict probability of voting for Trump (using logit model)
yhat_logit = cbind(pid = 0:6,
                   predict(l_reg,  # model object is different!
                           newdata = pred_dat, # data for prediction is the same!
                           type = "response")) %>%
  as.data.frame()
```

- By using the `type = "response"` option, we will obtain the predicted probabilities. However, the SE of the predicted probabilities is not available.
- SE is available, however, when we predict the *logit* by setting `se.fit = TRUE`. We specify the option `type = "link"` in the `predict` function. It can be shown that the sampling distribution of the predicted logits follow a Normal distribution in large samples
  - As the predicted logits are Normally distributed in large samples, we can use these estimated standard errors to calculate the 95% confidence intervals of the predicted logits. These intervals will have the form $\text{CI} = \text{logit}(\hat{\pi}_i) \pm 1.96 \times \widehat{\text{S.E.}}(\text{logit}(\hat{\pi}_i))$.
  - This will give us the confidence interval for the predicted logits. But we want the 95% CIs for the predicted probabilities. Here we use the fact that the inverse-logit function is a strictly increasing function and just apply the function to both end-points of the confidence interval. This will give us the confidence interval for the predicted probabilities. That is, if the 95% CI for the predicted logits has the form (a, b), then interval $(\text{logit}^{-1}(a), \text{logit}^{-1}(b))$ will be the 95% confidence interval for the predicted probabilities.
- In R, we can do this as follows:

```r
# predict the logit and standard errors
pred_logit <- predict(l_reg,
                      newdata = pred_dat,
                      response = "link",
                      se.fit = TRUE) %>%
    as.data.frame() %>%
    select(fit, se.fit)

# calculate 95% CI for logits
pred_logit <- pred_logit %>%
  mutate(lwr = fit - 1.96 * se.fit,
         upr = fit + 1.96 * se.fit)

# apply inverse-logit function to get pred. probs and CI
pred_p <- pred_logit %>%
  mutate_at(1:4, function(a){1 / (1 + exp(-a))}) %>%
  mutate(pid = pred_dat$pid)

# plot predicted probabilities (save plot in object l_plot)
l_plot <- pred_p %>%
  ggplot(aes(x = pid, y = fit)) +
  geom_line(col = "black") +
  geom_ribbon(aes(ymin = lwr, ymax = upr),
              fill = "grey", alpha = .5, col = NA) +
  scale_y_continuous(name = "Predicted Probability\n",
                     breaks = seq(0, 1, .25)) +
  scale_x_continuous(name = "\nParty Identification",
                     breaks = seq(0, 6, 1)) +
  geom_hline(yintercept = c(0, 1), linetype = 2) +
  geom_vline(xintercept = seq(0, 6, 1), linetype = 3, col = "grey") +
  theme_classic() +
```

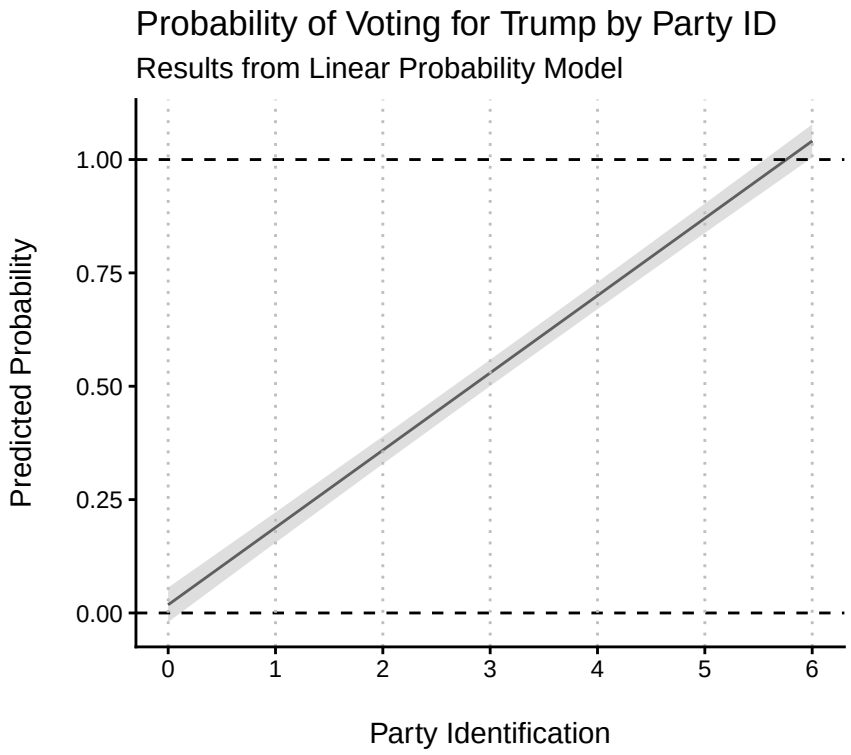
```
  ggtitle("Probability of Voting for Trump by Party ID",
          subtitle = "Results from Logistic Regression Model")

# print plot
print(l_plot)
```



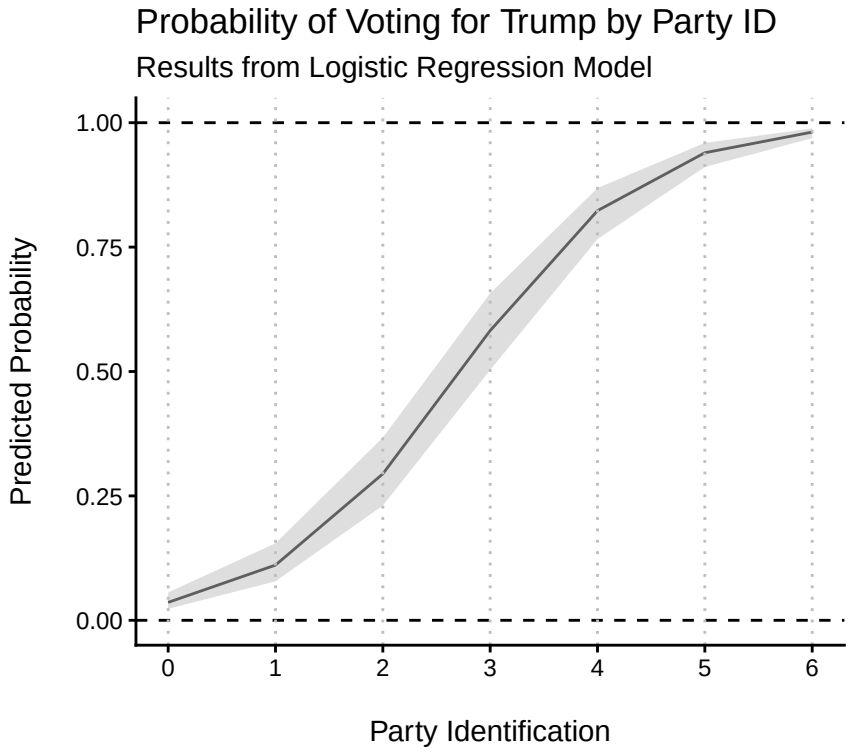
- Notice that all the predictions and the corresponding confidence intervals lie between zero and one, as desired. Furthermore, we see that our model predicts that the probability voting for Trump is almost zero for Strong Democrats (`pid = 0~1`) and almost one for Strong Republicans (`pid = 5~6`).
- This is a much more intuitive presentation of your results (or the meaning of the estimated regression coefficients) than an exponentiated coefficient of 3.341. So, whenever you run these models you should try to plot the predicted probabilities.

Lastly, we can compare the LPM and the logistic regression model:

```
lpm_plot
```




```
l_plot
```



3.7 Assumptions of Logistic Regressions

Some key assumptions for logistic regressions:

- Binary Y
- Observations are independent from each other
- Little or no multicollinearity between X variables
- X is linearly related to the *log odds* of Y

Do NOT need:

- Normally distributed errors
- Homoskedasticity (constant variance of errors)

Exercise 2 (20 minutes)

Were 1st class passengers more likely to survive on the Titanic? Using the `Titanic` data set (pre-loaded into R), run a logistic regression. How much did being in first class increase the odds of survival? Interpret your results (no need to calculate standard errors for this example), and then create a visualization if you have time!

```
# transforming the built-in Titanic data set into a tidy format
titanic <- Titanic %>%
  as_tibble() %>%
  uncount(weights = n) %>%
  clean_names()
```

```
# cleaning data for analysis
titanic_clean <- titanic %>%
  mutate(
    survived = ifelse(survived == "No", 0, 1)
  )

# fit logistic regression model (glm)
glm_titanic <- titanic_clean %>%
  glm(
    survived ~ class + sex + age, # formula of regression
    family = binomial(link = "logit"),        # specifying the dist. of outcome
    data = .                                  # data
)

# summarize results
summary(glm_titanic)
```

```
##
## Call:
## glm(formula = survived ~ class + sex + age, family = binomial(link = "logit"),
##     data = .)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.0438     0.1679  12.171  < 2e-16 ***
## class2nd     -1.0181     0.1960  -5.194 2.05e-07 ***
## class3rd     -1.7778     0.1716 -10.362  < 2e-16 ***
## classCrew    -0.8577     0.1573  -5.451 5.00e-08 ***
## sexMale      -2.4201     0.1404 -17.236  < 2e-16 ***
## ageChild      1.0615     0.2440   4.350 1.36e-05 ***
## ---
```

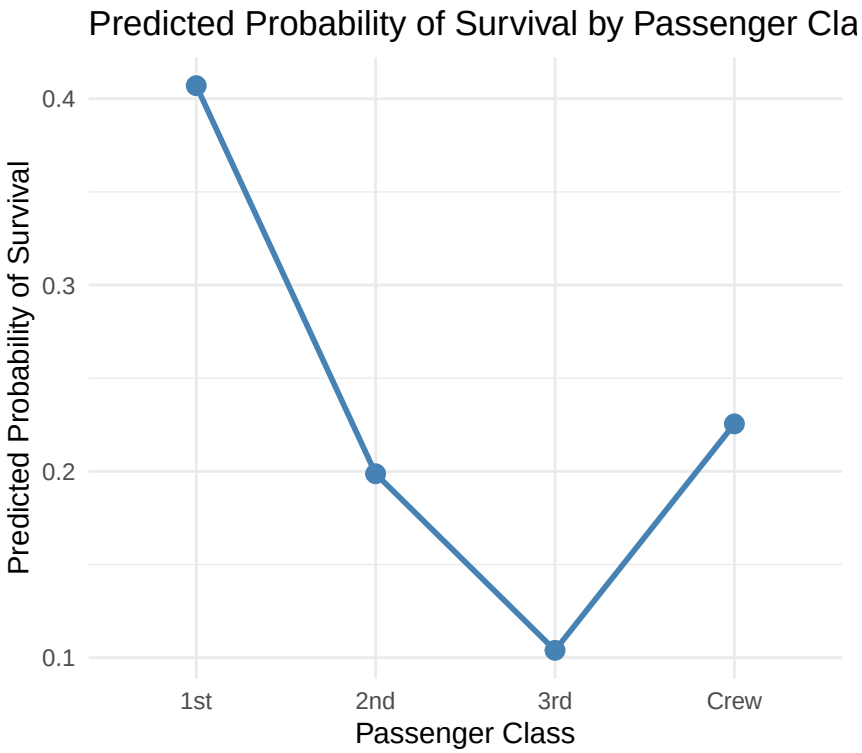
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2769.5  on 2200  degrees of freedom
## Residual deviance: 2210.1  on 2195  degrees of freedom
## AIC: 2222.1
##
## Number of Fisher Scoring iterations: 4
```

```
# extract coefficients
glm_coef = coef(glm_titanic)

# print coefficients and their exponentiated form
rbind(coef = glm_coef, exp_coef = exp(glm_coef))
```

```
##          (Intercept)   class2nd   class3rd  classCrew     sexMale ageChild
## coef        2.043837 -1.0180950 -1.7777622 -0.8576762 -2.42006035 1.061542
## exp_coef    7.720178  0.3612825  0.1690159  0.4241466  0.08891625 2.890826
```

```
# creating a prediction dataframe with fixed values
pred_dat <- data.frame(
  class = c("Crew", "3rd", "2nd", "1st"),
  sex = "Male",
  age = "Adult"
) %>%
  mutate(pred_prob = predict(glm_titanic, newdata = ., type = "response"))

ggplot(pred_dat, aes(x = class, y = pred_prob, group = 1)) +
  geom_line(color = "steelblue", size = 1) +
  geom_point(size = 3, color = "steelblue") +
  labs(
    title = "Predicted Probability of Survival by Passenger Class",
    x = "Passenger Class",
    y = "Predicted Probability of Survival"
  ) +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



We predict that moving from first class to second class means a passenger is .36 times as likely to survive, holding all other factors constant. Another way of saying this is that moving from first class to second class decreases the odds of survival by ~64%.