Supply Demand Gap Analysis

2024-09-28

#Purpose

The Purpose of this project is to provide background information to undergrad students, and to get a better understand of how ssaturated the labor market is for various degrees across the state of Ohio. Were you a student, this would be very useful information in deciding what university program to enter and where to focus your skill development. Were you to have individual-level demand data for each student with specific skills, you would have a much better idea of where you should place your own efforts to increase your earnings possibilities and hire-ability. As we only have access to public data, we will need to make some assumptions about supply by program, and demand by occupation. Namely, we are trusting the CIP-SOC crosswalk available on O*NET's site, which aligns Classification of Instructional Programs (CIPs) with Standard Occupational Classifications (SOCs). Of course, not all students who graduate from a specific program will be hired into a crosswalked occupation, but this crosswalk gives us general links between students and where they are qualified to be hired.

Overview

The **Supply-Demand Gap Analysis** consists of two parts: the supply and the demand. Demand data, specifically **LMI predicted yearly job openings**, are used to map demand by occupation. This RMarkdown outlines an attempt at modeling a supply demand gap analysis to help guide current student towards higher paying regions, and occupations with more demand relative to the number of graduates. This supply demand gap is calculated by dividing graduates sorted by **Classification of Instructional Programs (CIPs)** by job demand by **Standard Occupational Classifications (SOCs)**..

- **Supply Stock**: Represents the current available workforce for a given occupation or industry, including employed and unemployed individuals.
- **Supply Flow**: Represents the incoming workforce soon-to-be available, such as graduating students.

Data were updated to the most current versions and consolidated into tables. Specifically: - **IPEDS (Integrated Postsecondary Education Data System)** was updated through the academic year 2022, serving as the primary source for supply flow.

Data Cleaning and Transformation

Flow

- **IPEDS**: Publicly available at IPEDS Website, the list of CIPs includes all graduates in the state. Basic cleaning steps include:
  - Added new years of data up to 2022.
  - Standardized CIP codes and aggregated data by award level, program, and institution, then aggregating up to the JobsOhio region level and the state level.

Stock

- **LMI**: Long-term labor market information publicly available at LMI Website. Basic cleaning steps include:
  - Standardizing employment projections, occupational codes, and wage data format.

Data Structure

The counts by completers for the flow data source (IPEDS) are grouped as follows:

- **Year**: Aggregating data annually.
- **Graduating Program**: Using standardized CIP/SOC codes.
- **JobsOhio Region**: Adjustments based on zip code or county to JobsOhio region.
- **Education Classification**: Grouped into Certificate, Associate, Bachelor's, and Graduate levels.

Geography

- Adjusted based on zip code to county to JobsOhio region, and were downloaded from the Ohio Department of Jobs and Family Services. The zip to county data was found from the two sources below: HUD Office of Policy Development and Research and UnitedStatesZipCodes.org.

References

- **IPEDS**: IPEDS Data Center
- **OEWS**: BLS OEWS
- **LMI**: Ohio LMI
- **O*NET**: O*NET Online
- **Workforce Planning**: [Micheli, G. J. L., Martino, A., Porta, F., Cravello, A., Panaro, M., & Calabrese, A. (2023). Workforce planning in project-driven companies: a high-level guideline. *Frontiers in Industrial Engineering*, 1. https://doi.org/10.3389/fieng.2023.1267244


```
# This code reads the LMI jobs demand data by region and occupation from the ODJFS website.

#Paths
common_path <- getwd()
target_folder <- paste0(common_path, "/data/lmi-data/")

# Create the target folder, this will be helpful so all group members will automatically have a folder
  dir.create(target_folder, recursive = TRUE)
```

```
## Warning in dir.create(target_folder, recursive = TRUE):
## '/Users/stephenmonroe/Desktop/OSU Masters/BUSOBA
## 7250/7250-Project/data/lmi-data' already exists
```

```
# URLs for the different regions, these are all the excel sheets on the Ohio LMI website for each regio
url_northeast <- "https://ohiolmi.com/_docs/PROJ/JobsOhio/Northeast.xlsx"
url_central <- "https://ohiolmi.com/_docs/PROJ/JobsOhio/Central.xlsx"
url_west <- "https://ohiolmi.com/_docs/PROJ/JobsOhio/West.xlsx"
url_southeast <- "https://ohiolmi.com/_docs/PROJ/JobsOhio/Southeast.xlsx"
url_northwest <- "https://ohiolmi.com/_docs/PROJ/JobsOhio/Northwest.xlsx"
url_southwest <- "https://ohiolmi.com/_docs/PROJ/JobsOhio/Southwest.xlsx"



# Process Northeast region First
temp_northeast <- tempfile(fileext = ".xlsx")
response_northeast <- GET(url_northeast, write_disk(temp_northeast, overwrite = TRUE)) #Calls the url,
  headers_northeast <- suppressMessages(read_excel(temp_northeast, range = cell_rows(3:6)))
  #Have to get rid of bad headers
  headers_northeast <- apply(headers_northeast, 2, function(x) paste(na.omit(x), collapse = " "))
  headers_northeast <- c(headers_northeast, "med wage symbol")
```

```r
  data_northeast <- suppressMessages(read_excel(temp_northeast, skip = 5))
  #Skip the first 5 rows! all headers of white space.
  colnames(data_northeast) <- headers_northeast[1:12]
  #grab the names from only these headers
  rows_all_na_northeast <- rowSums(is.na(data_northeast)) == ncol(data_northeast)
  first_all_na_row_northeast <- which(rows_all_na_northeast)[1]
  data_northeast <- data_northeast[1:(first_all_na_row_northeast - 1), ]
  #that's annoying, but this should give us JUST the headers and not weird splits or missing headers.
  data_northeast$jobsohioregion <- "Northeast"


#OKAY, now do the same thing for all the other 5 regions, just past the above and change the region nam
# Process Central region_____________________________________________________________
temp_central <- tempfile(fileext = ".xlsx")
response_central <- GET(url_central, write_disk(temp_central, overwrite = TRUE))
  headers_central <- suppressMessages(read_excel(temp_central, range = cell_rows(3:6)))
  headers_central <- apply(headers_central, 2, function(x) paste(na.omit(x), collapse = " "))
  headers_central <- c(headers_central, "med wage symbol")
  data_central <- suppressMessages(read_excel(temp_central, skip = 5))
  colnames(data_central) <- headers_central[1:12]
  rows_all_na_central <- rowSums(is.na(data_central)) == ncol(data_central)
  first_all_na_row_central <- which(rows_all_na_central)[1]
    data_central <- data_central[1:(first_all_na_row_central - 1), ]
  data_central$jobsohioregion <- "Central"



# Process West region_____________________________________________________________
temp_west <- tempfile(fileext = ".xlsx")
response_west <- GET(url_west, write_disk(temp_west, overwrite = TRUE))
  headers_west <- suppressMessages(read_excel(temp_west, range = cell_rows(3:6)))
  headers_west <- apply(headers_west, 2, function(x) paste(na.omit(x), collapse = " "))
  headers_west <- c(headers_west, "med wage symbol")
  data_west <- suppressMessages(read_excel(temp_west, skip = 5))
  colnames(data_west) <- headers_west[1:12]
  rows_all_na_west <- rowSums(is.na(data_west)) == ncol(data_west)
  first_all_na_row_west <- which(rows_all_na_west)[1]
    data_west <- data_west[1:(first_all_na_row_west - 1), ]
  data_west$jobsohioregion <- "West"



# Process Southeast region_____________________________________________________________
temp_southeast <- tempfile(fileext = ".xlsx")
response_southeast <- GET(url_southeast, write_disk(temp_southeast, overwrite = TRUE))
  headers_southeast <- suppressMessages(read_excel(temp_southeast, range = cell_rows(3:6)))
  headers_southeast <- apply(headers_southeast, 2, function(x) paste(na.omit(x), collapse = " "))
  headers_southeast <- c(headers_southeast, "med wage symbol")
  data_southeast <- suppressMessages(read_excel(temp_southeast, skip = 5))
  colnames(data_southeast) <- headers_southeast[1:12]
  rows_all_na_southeast <- rowSums(is.na(data_southeast)) == ncol(data_southeast)
  first_all_na_row_southeast <- which(rows_all_na_southeast)[1]
    data_southeast <- data_southeast[1:(first_all_na_row_southeast - 1), ]
```

```r
  data_southeast$jobsohioregion <- "Southeast"


# Process Northwest region_____________________________________________________________________________
temp_northwest <- tempfile(fileext = ".xlsx")
response_northwest <- GET(url_northwest, write_disk(temp_northwest, overwrite = TRUE))
  headers_northwest <- suppressMessages(read_excel(temp_northwest, range = cell_rows(3:6)))
  headers_northwest <- apply(headers_northwest, 2, function(x) paste(na.omit(x), collapse = " "))
  headers_northwest <- c(headers_northwest, "med wage symbol")
  data_northwest <- suppressMessages(read_excel(temp_northwest, skip = 5))
  colnames(data_northwest) <- headers_northwest[1:12]
  rows_all_na_northwest <- rowSums(is.na(data_northwest)) == ncol(data_northwest)
  first_all_na_row_northwest <- which(rows_all_na_northwest)[1]
    data_northwest <- data_northwest[1:(first_all_na_row_northwest - 1), ]
  data_northwest$jobsohioregion <- "Northwest"



# Process Southwest region_____________________________________________________________________________
temp_southwest <- tempfile(fileext = ".xlsx")
response_southwest <- GET(url_southwest, write_disk(temp_southwest, overwrite = TRUE))
  headers_southwest <- suppressMessages(read_excel(temp_southwest, range = cell_rows(3:6)))
  headers_southwest <- apply(headers_southwest, 2, function(x) paste(na.omit(x), collapse = " "))
  headers_southwest <- c(headers_southwest, "med wage symbol")
  data_southwest <- suppressMessages(read_excel(temp_southwest, skip = 5))
  colnames(data_southwest) <- headers_southwest[1:12]
  rows_all_na_southwest <- rowSums(is.na(data_southwest)) == ncol(data_southwest)
  first_all_na_row_southwest <- which(rows_all_na_southwest)[1]
    data_southwest <- data_southwest[1:(first_all_na_row_southwest - 1), ]
  data_southwest$jobsohioregion <- "Southwest"

# Combine all region datasets into a single data frame
lmi_oews <- bind_rows(data_northeast, data_central, data_west, data_southeast, data_northwest, data_sou
#OKAY! all Regions loaded.
```

```r
#Ohio overall data_____________________________________________________________________________
# Define the column names manually, including the new 'median_wage_symbol'. This is because I cannot ge
column_names <- c(
  "soc_code",                    # SOC Code
  "soc_lmi_title",               # Occupational Title
  "employment",                  # Employment* 2020 Annual
  "projected_2030",              # 2030 Projected
  "change_employment",           # Change in Employment 2020-2030
  "percent_change",              # Percent
  "annual_openings_growth",      # Annual Openings Growth
  "exits",                       # Exits
  "transfers",                   # Transfers
  "total_openings",              # Total
  "median_wage",                 # Median Wage May 2021
  "median_wage_symbol",          # med wage symbol
  "Typical Education Needed for Entry",    # Not used in the select list
  "Work Experience in a Related Occupation",    # Not used in the select list
  "Typical On-The-Job Training Needed to Attain Competency" # Not used in the select list
)
```

```r
# Read the data from the Excel file, skipping the first three rows. I could not get the url to read in
ohio_data <- read_excel(paste0("./data/lmi-data/OccOH30_raw.xlsx"),
                        sheet = "Occupational Detail", skip = 3, col_names = FALSE)
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
## * `` -> `...15`
```

```r
ohio_data <- as.data.frame(ohio_data)
# Assign the manually defined column names to the data, these are defined above
colnames(ohio_data) <- column_names
# Add a new column 'jobsohioregion' with all values set to 'Ohio', this will give us the same manually
ohio_data <- ohio_data %>%
  mutate(jobsohioregion = 'Ohio')
```

```r
#Combine Ohio and Region Data_____________________________________________________________________________
# Ensure consistent column names and types for `ohio_data_trimmed`
ohio_data_trimmed <- ohio_data %>%
  select(
    soc_code, soc_lmi_title, employment, projected_2030,
    change_employment, percent_change, annual_openings_growth,
    exits, transfers, total_openings, median_wage,
    median_wage_symbol, jobsohioregion
  ) %>%
  mutate(
    employment = as.numeric(employment),  # Convert to numeric
    change_employment = as.numeric(change_employment),
    median_wage = as.numeric(median_wage),
    projected_2030 = as.numeric(projected_2030),
    percent_change = as.numeric(percent_change),
    annual_openings_growth = as.numeric(annual_openings_growth),
    exits = as.numeric(exits),
    transfers = as.numeric(transfers),
    total_openings = as.numeric(total_openings)
  )
```

```
## Warning: There were 9 warnings in `mutate()`.
## The first warning was:
## i In argument: `employment = as.numeric(employment)`.
## Caused by warning:
```

```
## ! NAs introduced by coercion
## i Run `dplyr::last_dplyr_warnings()` to see the 8 remaining warnings.
```

```
# Ensure column names and types match for `lmi_oews`
lmi_oews <- lmi_oews %>%
  rename(
    soc_code = `SOC Code`,
    soc_lmi_title = `Occupational Title`,
    employment = `Employment* 2020 Annual`,
    projected_2030 = `2030 Projected`,
    change_employment = `Change in Employment 2020-2030`,
    percent_change = `Percent`,
    annual_openings_growth = `Annual Openings Growth`,
    exits = `Exits`,
    transfers = `Transfers`,
    total_openings = `Total`,
    median_wage = `Median Wage May 2021`,
    median_wage_symbol = `med wage symbol`
  ) %>% mutate(
    employment = as.numeric(employment),  # Convert to numeric
    projected_2030 = as.numeric(projected_2030),
    change_employment = as.numeric(change_employment),
    percent_change = as.numeric(percent_change),
    annual_openings_growth = as.numeric(annual_openings_growth),
    exits = as.numeric(exits),
    transfers = as.numeric(transfers),
    total_openings = as.numeric(total_openings),
    median_wage = as.numeric(median_wage)
  )
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `median_wage = as.numeric(median_wage)`.
## Caused by warning:
## ! NAs introduced by coercion
```

```
# Ensure standardized column names for both data frames
colnames(ohio_data_trimmed) <- tolower(trimws(colnames(ohio_data_trimmed)))
colnames(lmi_oews) <- tolower(trimws(colnames(lmi_oews)))

# Combine the two datasets
ohio_region_lmi_data <- bind_rows(lmi_oews, ohio_data_trimmed)%>%
  mutate(
    jobsohioregion = case_when( #casewhen easiest in this case
      jobsohioregion == "Northwest" ~ 1L,
       jobsohioregion == "West" ~ 2L,
      jobsohioregion == "Southwest" ~ 3L,
      jobsohioregion == "Northeast" ~ 4L,
      jobsohioregion == "Central" ~ 5L,
      jobsohioregion == "Southeast" ~ 6L,
      jobsohioregion == "Ohio" ~ 39L, #ohio to 39, check this is true for all
      TRUE ~ NA_integer_  # For any unmatched regions, set to NA, should removed these or see why they
     )
    )
#Will have to fix manual vs hourly wage data later on it looks like. Pay attention to the wage symbol.
#SAVE the data
```

```r
rda_file_path <- paste0(target_folder, "ohio_region_lmi_data.rda") #rda's always better (I think?)
save(ohio_region_lmi_data, file = rda_file_path)
```

```r
# IPEDS Directory data ----- Check the Read_me file in the data folder for instructions on how to pull.
# https://nces.ed.gov/ipeds/use-the-data
# Get JOR codes to attach to the IPEDS directory data
load('data/cross-walks/jobsohioregions.rda')

ipeds_directory <- read_csv('data/ipeds-institution-detail/STATA_RV_7162021-493.zip') %>%
  left_join(jobsohioregions, by = c('countycd' = 'statefips')) %>%
  transmute(
    ipeds_code = unitid,
    institutionname = instnm,
    street_address = addr,
    city = city,
    state = stabbr,
    zip = zip,
    web_address = webaddr,
    regionId = jobsohioregion,
    lat = latitude,
    lng = longitud
  )
```

```
## Multiple files in zip: reading 'STATA_RV_7162021-493.csv'
## Rows: 296 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): instnm, ialias, addr, city, stabbr, zip, webaddr, ein, countynm, c...
## dbl (13): unitid, year, gentele, countycd, longitud, latitude, newid, deathy...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
save(ipeds_directory, file = 'data/ipeds-institution-detail/ipeds_directory.rda')
```

```r
# IPEDS Data ----- Check the Read_me file in the data folder for instructions on how to pull.
# https://nces.ed.gov/ipeds/use-the-data

# label values cipcode         label_cipcode
# label define label_awlevel         15 "Degrees/certificates total"
# label define label_awlevel         12 "Degrees total", add
# label define label_awlevel         3 "Associate''s degree", add
# label define label_awlevel         5 "Bachelor''s degree", add
# label define label_awlevel         7 "Master''s degree", add
# label define label_awlevel         9 "Doctor''s degree", add
# label define label_awlevel         10 "First-professional degree", add
# label define label_awlevel         13 "Certificates below the bacculaureate total", add
# label define label_awlevel         1 "Award of less than 1 academic year", add
# label define label_awlevel         2 "Award of at least 1 but less than 2 academic years", add
# label define label_awlevel         4 "Award of at least 2 but less than 4 academic years", add
# label define label_awlevel         14 "Certificates above the bacculaureate total", add
# label define label_awlevel         6 "Postbaccalaureate certificate", add
# label define label_awlevel         8 "Post-master''s certificate", add
# label define label_awlevel         11 "First-professional certificate", add
```

```r
#Using the above category definitions from the STATA file you can download from IPEDS, let's remap to l
#so we actually know what is going on
ipeds_degree_remapping <- tribble(
  ~awlevel, ~degree_group_logord,
  '1',        1L,
  '2',        1L,
  '3',        2L,
  '4',        1L,
  '5',        3L,
  '6',        1L,
  '7',        4L,
  '8',        5L,    # grad certificate, has not been included in the Supply Tool
  '9',        4L,
  '10',         4L,
  '11',         5L,    # grad certificate, has not been included in the Supply Tool
  '12',       NA,    # subtotals
  '13',       NA,    # subtotals
  '14',       NA,    # subtotals
  '15',       NA,    # subtotals
  '17',         4L,
  '18',         4L,
  '19',         4L
)

# Read files, keep only 6-digit CIP, address some variable name changes (crace24/ctotalt)
# Using default character because it is easier to start from there, keep CIP codes correct,

#First, use list.files to find the .zip files that download from IPEDS, better to store them as .zip, b
ipeds_completions <- list.files('data/ipeds-completions', '.*zip$', full.names = TRUE) %>%
  map_dfr(~ read_csv(., col_types = cols(.default = col_character()))) %>%
  filter(nchar(cipcode) == 7) %>%  # 7 because of the "." in the number, e.g. "15.0101"
  mutate(grads = as.integer(ctotalt)) %>% #this is the grads count column
  left_join(ipeds_degree_remapping, by = 'awlevel') %>%
  filter(!is.na(degree_group_logord) & grads > 0) %>% # drop subtotals and zero rows
  group_by(unitid, year, cipcode, degree_group_logord) %>%  # this is for combining majornum = 1 and ma
  summarise(graduates = sum(grads), .groups = 'drop') %>%
  left_join(transmute(ipeds_directory, unitid = as.character(ipeds_code), regionId), by = 'unitid') %>%
  select(ipeds_code = unitid,
         cip_code = cipcode,
         degree_group_logord,
         academic_year = year,
         jobsohioregion = regionId,
         graduates)
```

```
## Multiple files in zip: reading 'STATA_RV_3172022-1009.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-1030.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-141.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-185.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-301.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-502.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-620.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-893.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-949.csv'
## Multiple files in zip: reading 'STATA_RV_3172022-974.csv'
```

```
## Multiple files in zip: reading 'STATA_RV_582024-207.csv'
## Multiple files in zip: reading 'STATA_RV_962022-18.csv'
```

```
save(ipeds_completions, file = 'data/ipeds-completions/ipeds_completions.rda')
```

##End OF data Import, now need to Combine according to CIP-SOC Crosswalk

Final Datasets Created:

ohio_region_lmi_data: Occupation demand dataset that includes six Ohio regions and statewide data (jobsohioregion coded numerically for each region).

Main Variables: -soc_code: Standard Occupational Classification code. -soc_lmi_title: Occupation title based on LMI. -employment: Employment count for 2020. -projected_2030: Projected employment count for 2030. -change_employment: Change in employment from 2020 to 2030. -percent_change: Percentage change in employment. -annual_openings_growth: Annual growth in job openings. -median_wage: Median wage in 2021. -jobsohioregion: Region identifier (1-6 for regions, 39 for Ohio).

ipeds_completions.rda:IPEDS completions data for institutions in Ohio, linked to LMI regions.

Main Variables: -ipeds_code: Unique identifier for institutions. -cip_code: Classification of Instructional Programs code for program areas. -degree_group_logord: Ordinal representation of degree levels (e.g., 1 for -certificates, 2 for associate degrees, 3 for bachelor's degrees). -academic_year: Year of data collection. -jobsohioregion: Region identifier linked to LMI regions. -graduates: Number of graduates in a given program and year.

```
#Adjust the yearly median wage to hourly from LMI
# Convert median wage from yearly to hourly if the symbol is "**". It's weird and there is probably
#a more efficient way to do this, but I am just using mutate and gsub for each case. If it is a yearly
#symbol, I am calculating hourly wage from yearly by assuming 2080 hours in the year.
lmi_oews <- ohio_region_lmi_data%>%
  mutate(jobsohioregion = as.character(jobsohioregion))%>%
  mutate(median_wage = gsub("[^0-9.]", "", median_wage))%>%
  mutate(median_wage_symbol = gsub("[^0-9.]", "", median_wage_symbol))%>%
  mutate(median_wage = as.numeric(median_wage)) %>%
  mutate(median_wage_symbol = trimws(median_wage_symbol))%>%
  mutate(
    median_wage = case_when(
       !is.na(median_wage_symbol) & median_wage_symbol == "**" ~ median_wage / 2080,# Convert from year
      !is.na(median_wage_symbol) & median_wage_symbol == "††" ~ median_wage / 2080, # Convert from stat
      !is.na(median_wage_symbol) & median_wage_symbol == "†" ~ median_wage,# Statewide hourly wage ('†'
      !is.na(median_wage_symbol) & median_wage_symbol == " " ~ NA_real_,  # Wage not available (' '), se
      median_wage >= 1000 ~ median_wage / 2080, #final check for over $1000 an hour, we maybe should ju
      TRUE ~ median_wage  # Keep as is for other cases
    )
  )
#run it and it looks like for our data, 96$ an hour is the max, makes sense. Those making much more #pr

summary(lmi_oews$median_wage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   9.238  17.900  23.090  26.684  30.863  96.981      27
```

The lowest median hourly range is around 9.39 and the max is 96.98. The median hourly wage is 23.09 and the mean is 26.68. Since the median is lower than the mean, this means that there are some higher wages that is pulling the mean up.

```
library(dplyr)
```

```r
# Combine and De-duplicate Graduate Data ____________________________________
graduates_data <- ipeds_completions%>%
  dplyr::filter(academic_year >= 2010) %>% #here, academic year is the regular school year, so around S
  mutate(academic_year = as.integer(academic_year), jobsohioregion = as.character(jobsohioregion))

#Aggregate Data by Region and State_____________________________________________
# Summarize graduates by region
graduates_regions <- graduates_data %>%
  group_by(jobsohioregion, cip_code, degree_group_logord, academic_year)%>%
  #this is a count of graduates for each program, for each degree type, for each region, for each year.
  summarise(graduates = sum(graduates, na.rm = TRUE), .groups = "drop")

# Summarize graduates for the entire state, so same as prior chunk, but for the state overall
graduates_statewide <- graduates_regions %>%
  group_by(cip_code, degree_group_logord, academic_year) %>%
  summarise(graduates = sum(graduates, na.rm = TRUE), .groups = "drop") %>%
  mutate(jobsohioregion = "39")

# Combine regional and statewide data
state_region_graduates <- bind_rows(graduates_regions, graduates_statewide)
rm(graduates_regions, graduates_statewide) #don't save the old versions
```

```r
#read in crosswalk and do second sheet, which is CIP-SOC
cip_soc <- read_excel("data/cross-walks/CIP2020_SOC2018_Crosswalk.xlsx", sheet = 'CIP-SOC')
soc_cip <- read_excel("data/cross-walks/CIP2020_SOC2018_Crosswalk.xlsx", sheet = 'SOC-CIP')


# Join Graduate Data with CIP-SOC Mappings_____________________________________________________
#Standardize column names in both data sets
cip_soc <- cip_soc %>%
  mutate(cip_code = trimws(CIP2020Code))%>%
  mutate(soc_code = trimws(SOC2018Code))
soc_cip <- soc_cip %>%
  mutate(cip_code = trimws(CIP2020Code))%>%
  mutate(soc_code = trimws(SOC2018Code))

state_region_graduates <- state_region_graduates %>%
  mutate(cip_code = trimws(cip_code))

#Check for unmatched `cip_code` values before joining
unmatched_cip_codes <- setdiff(state_region_graduates$cip_code, cip_soc$cip_code)
print(unmatched_cip_codes)  # Check for missing or mismatched `cip_code` values
```

```
##  [1] "15.0503" "15.0505" "43.0106" "43.0111" "43.0116" "43.0117" "51.0808"
##  [8] "51.1104" "51.2501" "51.3817" "43.0118" "01.0309" "19.0000" "51.2101"
## [15] "51.2401" "51.1901"
```

```r
#merge the SOC codes into our graduate data, so we have counts by all CIP-SOC matchings. If we want to
# for a specific SOC, we can sum up the graduates grouped by soc, degree, year, region.
#IMPORTANT TO REMEMBER, AFTER THIS STEP THEY ARE NO LONGER UNIQUE COUNTS, BUT MUST BE INTERPRITED BY TH
aggregated_data <- state_region_graduates %>%
  mutate(cip_only_id = paste(cip_code, degree_group_logord, academic_year, jobsohioregion, sep = "_")) %
  left_join(cip_soc, by = c("cip_code" = "cip_code")) %>% #join in our SOC codes, most CIP codes match
    mutate(cip_code_soc_code_filter_groups_id = paste(cip_code, soc_code, degree_group_logord, academic_
```

```
  group_by(soc_code, cip_code, degree_group_logord, jobsohioregion, academic_year) %>%
  summarise(total_graduates = sum(graduates, na.rm = TRUE), .groups = "drop") %>%
  #For each CIP-SOC match-up, we have total-graduates.
   dplyr::select(cip_code, academic_year, jobsohioregion, degree_group_logord, total_graduates, soc_code
```

```
## Warning in left_join(., cip_soc, by = c(cip_code = "cip_code")): Detected an unexpected many-to-many
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 6 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
nrow(aggregated_data)
```

```
## [1] 253591
```

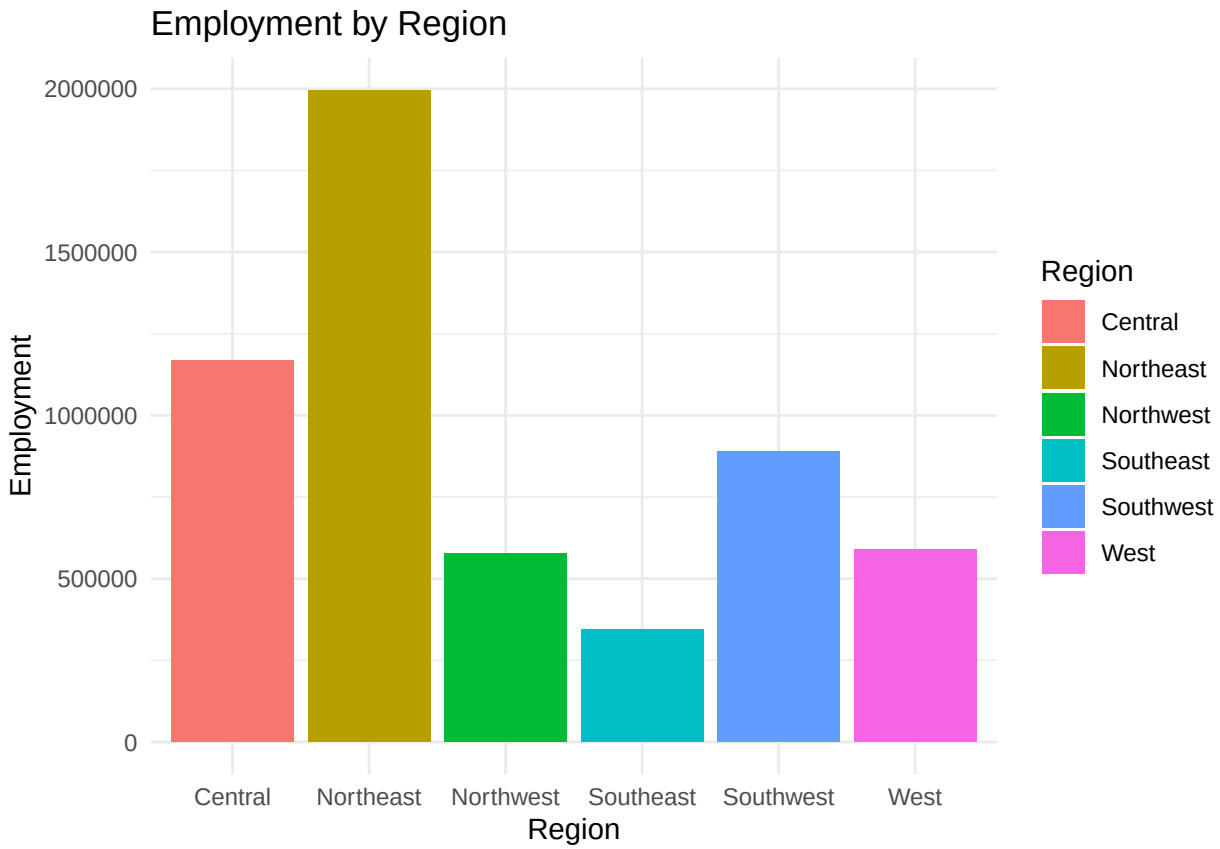
```
#As an exercise, summarize JUST by our unique id variable, and see if we get the same number of observa
# aggregated_dataII <- state_region_graduates %>%
#   mutate(cip_only_id = paste(cip_code, degree_group_logord, academic_year, jobsohioregion, sep = "_"))
#   left_join(cip_soc, by = c("cip_code" = "cip_code")) %>%
#     mutate(cip_code_soc_code_filter_groups_id = paste(cip_code, soc_code, degree_group_logord, academ
#   group_by(cip_code_soc_code_filter_groups_id) %>%
#   summarise(total_graduates = sum(graduates, na.rm = TRUE), .groups = "drop") %>%
#    select(cip_code_soc_code_filter_groups_id, total_graduates)
# nrow(aggregated_dataII)


# Calculate Total CIP Graduates per SOC, Region, and Degree Group _____________________________________
total_cip_graduates_per_soc <- aggregated_data %>%
  group_by(soc_code, jobsohioregion, degree_group_logord, academic_year)%>%
   summarise(total_cip_graduates_by_soc = sum(total_graduates, na.rm = TRUE), .groups = "drop")%>%
  dplyr::filter(!is.na(jobsohioregion))
#So the graduate counts in this table represent all graduates in the same region, and academic year who
#available to work in each occupation, separated by degree type.

# Merge total CIP graduates back with the main data
aggregated_data <- aggregated_data %>%
  left_join(total_cip_graduates_per_soc, by = c("soc_code", "jobsohioregion", "degree_group_logord", "ac

# Integrate LMI Data  _____________________________________________
aggregated_data_with_lmi <- aggregated_data %>%
  left_join(lmi_oews, by = c("soc_code", "jobsohioregion"))#the lmi_oews data applies to all years, we
  # mutate(adjusted_demand = annual_openings_growth * (as.numeric(total_graduates) / as.numeric(total_c
```

```
#lets do the gap ratio calculations
# Calculate Gap Ratio _____________________________________________________________________
master_aggregated_data <- aggregated_data_with_lmi%>%
  group_by(soc_code, cip_code, jobsohioregion, academic_year)%>% #took off CIP code
  summarise(

                        ####GAP RATIO CALCULATION BELOW#####
        gap_ratio  = sum(total_graduates) / sum(annual_openings_growth, na.rm = TRUE),
                        ####################################

    # Preserve columns by taking their first occurrence. They are all the same, but I forget the correc
    #this fact so they are preserved....
```

```
    total_cip_graduates_by_soc = first(total_cip_graduates_by_soc),
    total_graduates = first(total_graduates),
    employment = first(employment),
    annual_openings_growth = first(annual_openings_growth),
    median_wage = first(median_wage))%>%
  #And finally, rename the regions for our visualizations!
  mutate(jobsohioregion = case_when(
    jobsohioregion == '1' ~ 'Northwest',
    jobsohioregion == '2' ~ 'West',
    jobsohioregion == '3' ~ 'Southwest',
    jobsohioregion == '4' ~ 'Northeast',
    jobsohioregion == '5' ~ 'Central',
    jobsohioregion == '6' ~ 'Southeast',
    jobsohioregion == '39' ~ 'Ohio',
    TRUE ~ as.character(jobsohioregion)
  ))
```

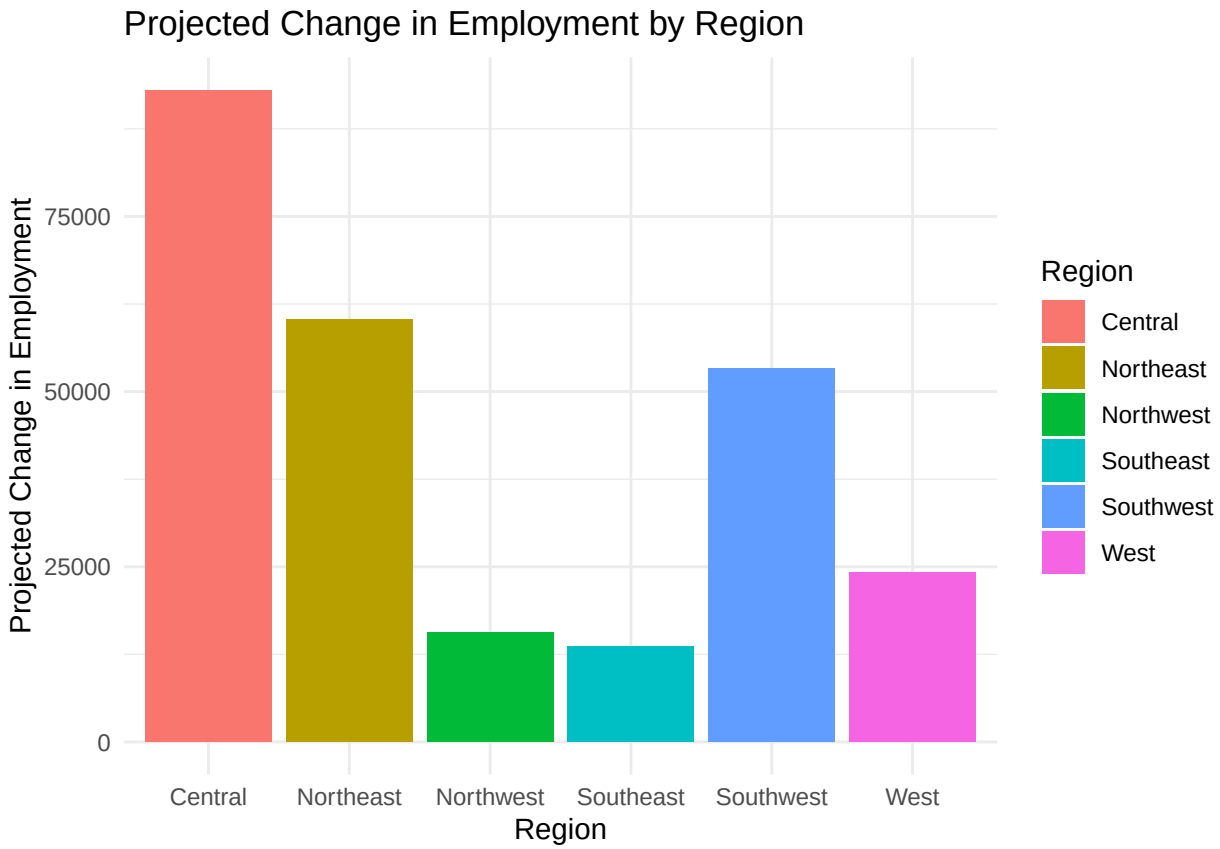
```
## `summarise()` has grouped output by 'soc_code', 'cip_code', 'jobsohioregion'.
## You can override using the `.groups` argument.
```

```
ohio_region_lmi_data %>%
  mutate(jobsohioregion = case_when(
    jobsohioregion == '1' ~ 'Northwest',
    jobsohioregion == '2' ~ 'West',
    jobsohioregion == '3' ~ 'Southwest',
    jobsohioregion == '4' ~ 'Northeast',
    jobsohioregion == '5' ~ 'Central',
    jobsohioregion == '6' ~ 'Southeast',
    jobsohioregion == '39' ~ 'Ohio',
    TRUE ~ as.character(jobsohioregion))) %>%
  filter(jobsohioregion != "Ohio" & soc_lmi_title == "Total, All Occupations") %>%
  ggplot(aes(x = factor(jobsohioregion), y = employment, fill = jobsohioregion)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Employment by Region",
       x = "Region",
       y = "Employment",
       fill = "Region") +
  theme_minimal()
```



The above chart shows what the total employment is per region in Ohio. The northeast region has the most people employed. The southeast region has the lowest number of people employed.

```
ohio_region_lmi_data %>%
  mutate(jobsohioregion = case_when(
    jobsohioregion == '1' ~ 'Northwest',
    jobsohioregion == '2' ~ 'West',
    jobsohioregion == '3' ~ 'Southwest',
    jobsohioregion == '4' ~ 'Northeast',
    jobsohioregion == '5' ~ 'Central',
    jobsohioregion == '6' ~ 'Southeast',
    jobsohioregion == '39' ~ 'Ohio',
    TRUE ~ as.character(jobsohioregion))) %>%
  filter(jobsohioregion != "Ohio" & soc_lmi_title == "Total, All Occupations") %>%
  ggplot(aes(x = factor(jobsohioregion), y = change_employment, fill = jobsohioregion)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Projected Change in Employment by Region",
       x = "Region",
       y = "Projected Change in Employment",
       fill = "Region") +
  theme_minimal()
```



This chart shows the projected change in number of jobs in each region between now and 2030. All regions are projected to grow. The central will grow the most. The northwest and southeast will have the least, with the southeast being the lowest.

```
lmi_oews %>%
  mutate(jobsohioregion = case_when(
    jobsohioregion == '1' ~ 'Northwest',
    jobsohioregion == '2' ~ 'West',
    jobsohioregion == '3' ~ 'Southwest',
    jobsohioregion == '4' ~ 'Northeast',
    jobsohioregion == '5' ~ 'Central',
    jobsohioregion == '6' ~ 'Southeast',
    jobsohioregion == '39' ~ 'Ohio',
    TRUE ~ as.character(jobsohioregion))) %>%
  filter(jobsohioregion != "Ohio" & soc_lmi_title != "Total, All Occupations") %>%
  ggplot(aes(x = annual_openings_growth, y = median_wage, color = factor(jobsohioregion))) +
  geom_point() +
  labs(title = "Median Hourly Wage vs. Annual Openings Growth",
       x = "Annual Openings Growth",
       y = "Median Hourly Wage",
       color = "Region") +
  theme_minimal()
```

```
## Warning: Removed 16 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

This plot is a scatter that compares the annual jobs growth against what the median hourly wage is for that job. There doesn't appear to be a trend between growth and wage. The majority of growth appears to be between an annual loss or gain of 500 jobs.

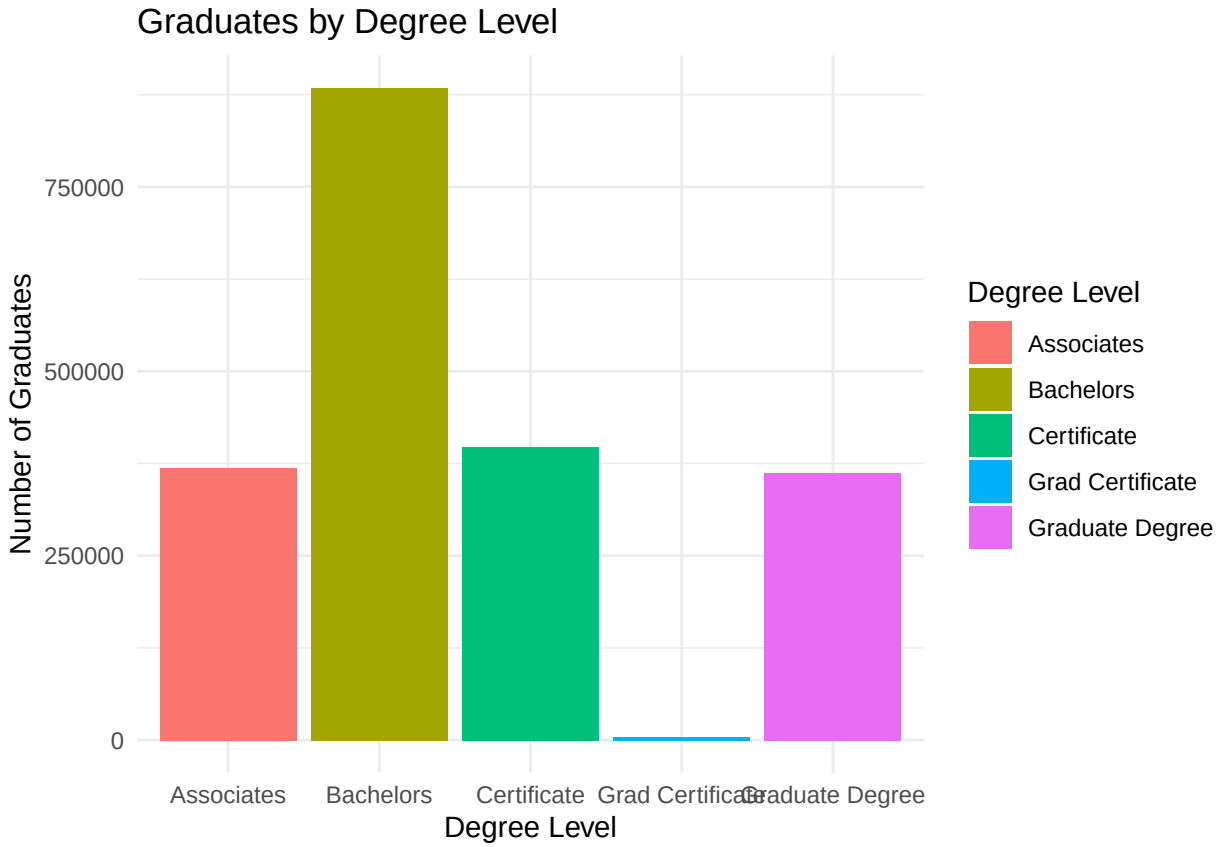
```
lmi_oews %>%
  mutate(jobsohioregion = case_when(
    jobsohioregion == '1' ~ 'Northwest',
    jobsohioregion == '2' ~ 'West',
    jobsohioregion == '3' ~ 'Southwest',
    jobsohioregion == '4' ~ 'Northeast',
    jobsohioregion == '5' ~ 'Central',
    jobsohioregion == '6' ~ 'Southeast',
    jobsohioregion == '39' ~ 'Ohio',
    TRUE ~ as.character(jobsohioregion))) %>%
  filter(jobsohioregion != "Ohio" & soc_lmi_title != "Total, All Occupations") %>%
  filter(jobsohioregion == "Northeast" & (annual_openings_growth >= 1750 | annual_openings_growth <= -1
  select(soc_lmi_title, annual_openings_growth) %>%
  print()
```

```
## # A tibble: 2 x 2
##   soc_lmi_title                                    annual_openings_growth
##   <chr>                                                             <dbl>
## 1 Food Preparation and Serving Related Occupations                   1866
## 2 Office and Administrative Support Occupations                     -1457
```

Based on the previous plot, we could see that the most and least job growth was in the Northeast region. We could also tell that the growth was more than 1750 and the decrease was more than 1000. Using the region and annual growth, we can find that the largest growth was in the food prep and serving space and

the largest decline was in the office and admin support space.

```
ipeds_completions %>%
  mutate(degree_group_logord = case_when(
    degree_group_logord == 1 ~ 'Certificate',
    degree_group_logord == 2 ~ 'Associates',
    degree_group_logord == 3 ~ 'Bachelors',
    degree_group_logord == 4 ~ 'Graduate Degree',
    degree_group_logord == 5 ~ 'Grad Certificate')) %>%
  ggplot(aes(x = factor(degree_group_logord), y = graduates, fill = factor(degree_group_logord))) +
  geom_bar(stat = "identity") +
  labs(title = "Graduates by Degree Level",
       x = "Degree Level",
       y = "Number of Graduates",
       fill = "Degree Level") +
  theme_minimal()
```

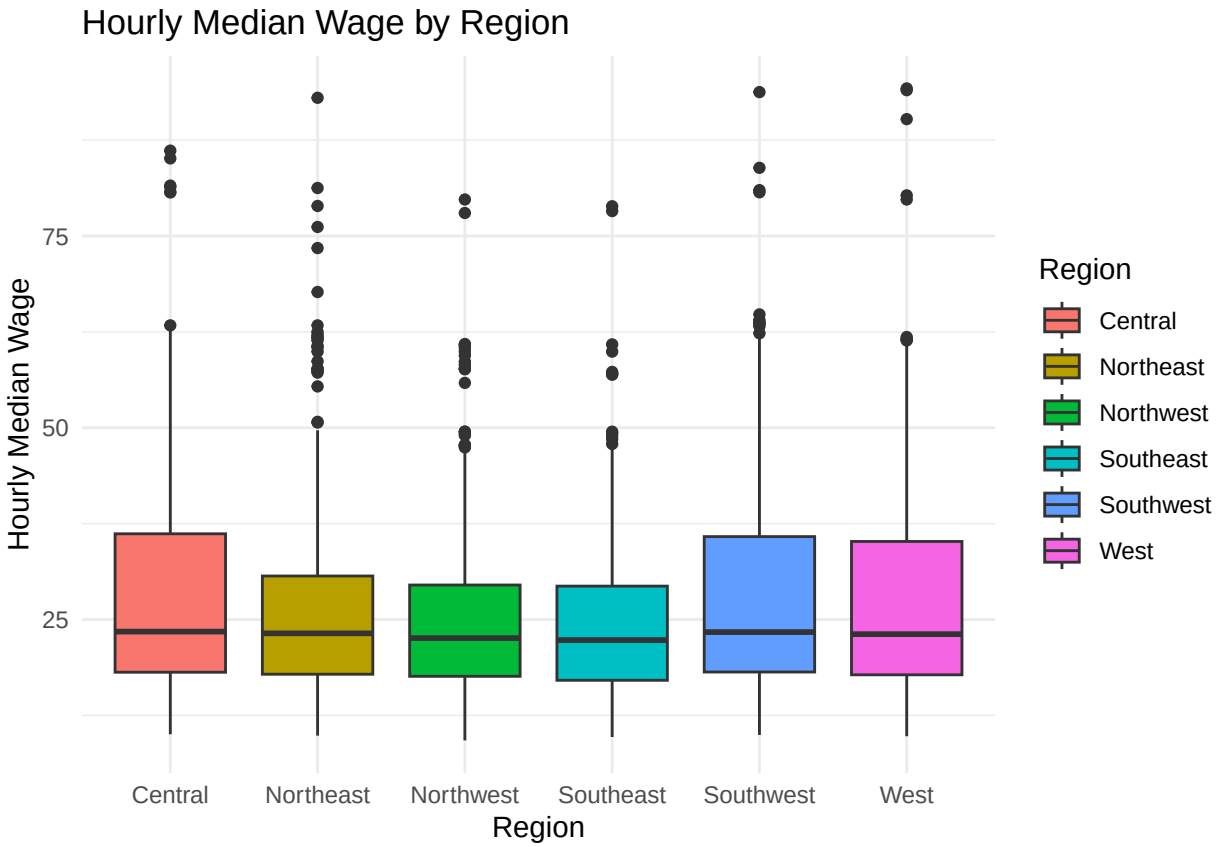


This plot shows the number of graduates that have varying degrees. The largest degree is the bachelors and the lowest is the grad certificate. All other degrees have similar numbers of graduates across them.

```
lmi_oews %>%
  mutate(jobsohioregion = case_when(
    jobsohioregion == '1' ~ 'Northwest',
    jobsohioregion == '2' ~ 'West',
    jobsohioregion == '3' ~ 'Southwest',
    jobsohioregion == '4' ~ 'Northeast',
    jobsohioregion == '5' ~ 'Central',
    jobsohioregion == '6' ~ 'Southeast',
```

```
    jobsohioregion == '39' ~ 'Ohio',
    TRUE ~ as.character(jobsohioregion))) %>%
  filter(jobsohioregion != "Ohio" & soc_lmi_title != "Total, All Occupations") %>%
  ggplot(aes(x = factor(jobsohioregion), y = median_wage, fill = factor(jobsohioregion))) +
  geom_boxplot() +
  labs(title = "Hourly Median Wage by Region",
       x = "Region",
       y = "Hourly Median Wage",
       fill = "Region") +
  theme_minimal()
```

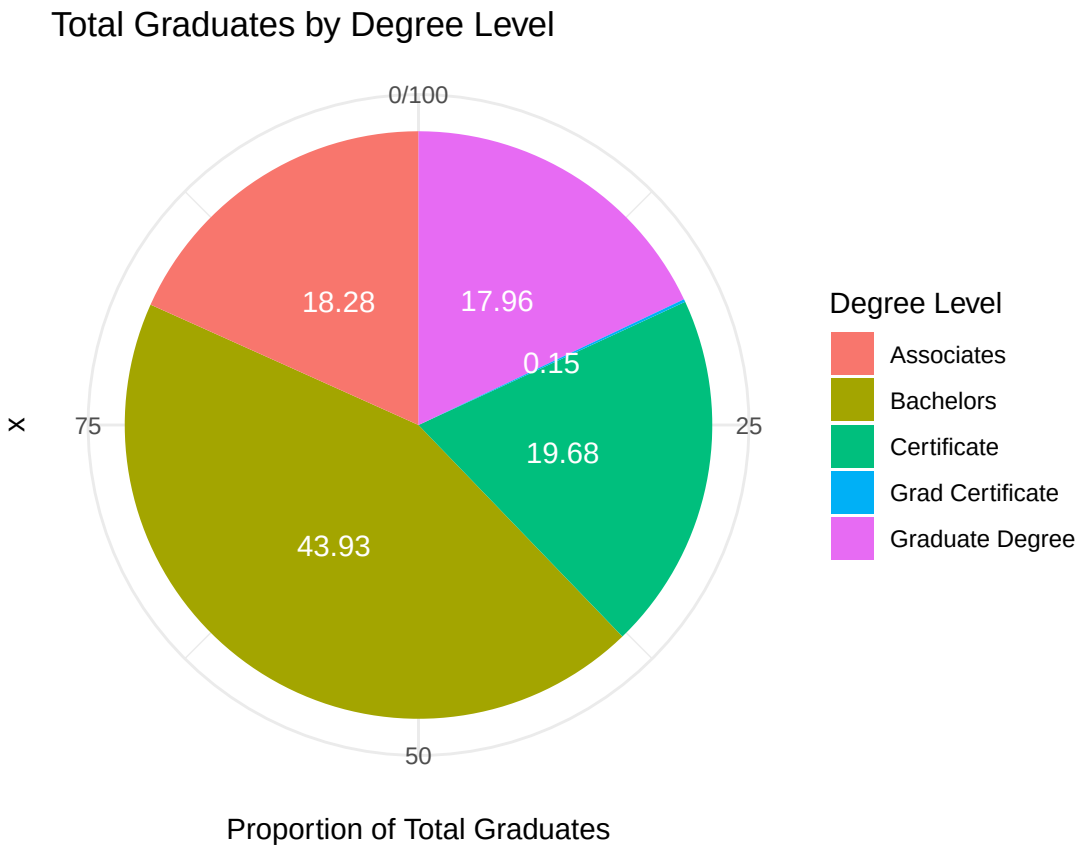
```
## Warning: Removed 16 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



This plot is a box and whisker of hourly median wage and broken up by region. Each region appears to have around the same median wage. The Central, Southwest, and West appear to have similar 25th to 75th percentiles and the the Northeast, Northwest, Southeast have the same thing between them. Each region does have outlier median hourly wages.

```
total_graduates_by_degree <- ipeds_completions %>%
  mutate(degree_group_logord = case_when(
    degree_group_logord == 1 ~ 'Certificate',
    degree_group_logord == 2 ~ 'Associates',
    degree_group_logord == 3 ~ 'Bachelors',
    degree_group_logord == 4 ~ 'Graduate Degree',
    degree_group_logord == 5 ~ 'Grad Certificate')) %>%
  group_by(degree_group_logord) %>%
  summarise(total_graduates = sum(graduates, na.rm = TRUE)) %>%
```

```
  arrange(desc(total_graduates)) %>%
  mutate(prop = round(total_graduates / sum(total_graduates) *100, digits = 2))

ggplot(total_graduates_by_degree, aes(x = "", y = prop, fill = factor(degree_group_logord))) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  labs(title = "Total Graduates by Degree Level",
       fill = "Degree Level",
       y = "Proportion of Total Graduates") +
  geom_text(aes(label = prop),
            color = "white",
            position = position_stack(vjust = 0.5)) +
  theme_minimal()
```



This pie chart breaks down the total number of graduates into what degree they have. It shows that almost 50% have a bachelors and less than 1% have a graduate certificate.