Assesssment 1 - COVID-19 impact on digital learning

Nguyen Khuat Son Tra and 48144134

2024-04-11

Over the three years since the onset of COVID-19 lockdowns at the beginning of 2020, significant changes have unfolded. One notable transformation is the prominent role of online learning in the educational landscape. While face-to-face classes continues to coexist, online platforms have become integral in enhancing the overall learning experience.

Upon reflecting on three major themes emerging one year after the onset of COVID-19 – namely, the exacerbation of pre-existing inequalities and shortcomings in the educational system, the advocacy for transformative learning, and the demand for innovative approaches in community and intergenerational learning (Stanistreet et al., 2020) – the dataset examined in this report will offer insights into the impact of COVID-19 on learning platforms across various regions in the United States.

1. Data Cleaning and Wrangling

To begin, we set up the working directory and import necessary library to prepare for the work.

1.1 Data Cleaning

Respectively, we will clean 5 engagement files (1000.csv, 1039.csv, 1044.csv, 1052.csv, and 1131.csv), products_info.csv and districts_info.csv).

Context

After reviewing all engagement datasets, we have decided to exclude the dataset “1131.csv” to reduce noise in the dataset. While both “1131.csv” and “1039.csv” lack district information, we observed that while “1039.csv” is marked as “NaN”, the state for “1131.csv” is “don92t know”. Given the guidelines, “NaN” indicates that these values were intentionally suppressed to enhance dataset anonymization. By retaining one representative of the “NaN” state in our analysis, we maintain dataset integrity without introducing excessive noise.

Engagement Data

Between 1.01.2020 (few months before the pandemic outbreak), and 31.12.2020, five engagement files were observed. Each file within the engagement data captured various product IDs used during the mentioned period within a single district.

a. 1000.csv (Connecticut). To begin, we observe the dataset: 1000.csv by examining a few initial rows and their columns.

```
engage1 = read_csv(file="Engagement Data/1000.csv") #rename for clarity

## Warning: One or more parsing issues, call `problems()` on your data frame
for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)

## Rows: 104003 Columns: 4
## ── Column specification
────────────────────────────────────────────────────────

## Delimiter: ","
## chr (1): time
## dbl (3): lp_id, pct_access, engagement_index
##
## 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.

colSums(is.na(engage1)) # check n/a in each column

##             time            lp_id       pct_access engagement_index
##                0                2                1            42348
```

In the next step, we start to look into each column (time, lp_id, pct_access, and engagement_index).

Every time we clean, the ‘unique()’ function was used initially to observe the data and later to verify if changes were made. However, due to word limitations, we do not include this information in the following.

```
  ## transform the data: time and lp_id column
engage1 <- engage1 %>%
  mutate(time = replace(time, time == "1/01/2022", "1/01/2020")) %>%
  mutate(time = replace(time, time == "31/12/1020", "31/12/2020")) %>%
  filter(!is.na(lp_id)) # dropping n/a

colSums(is.na(engage1)) # verify if n/a has been filtered

##             time            lp_id       pct_access engagement_index
##                0                0                1            42346
```

Given the unique nature of lp_id, individually checking all 104,003 observations is not efficient. Hence, we only clean and drop n/a values for the first two columns to avoid inadvertently removing necessary time and lp_id data, vital for later merging and visualization. Notably, the 'engagement_index' column contains 42,346 n/a values, almost half the dataset. Thus, removing them all would compromise the integrity of 'time' and 'lp_id' information.

Finally, for the wrangling process, we add an column named 'district_id'.

```
# add district_id column to prepare for the merging step
engage1 <- mutate(engage1, district_id = 1000)
```

b. 1039.csv (NaN). Similarly, we repeat the process for 1039.csv.

```
engage2 = read_csv(file="Engagement Data/1039.csv") #rename for clarity

## Rows: 38791 Columns: 4
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr (2): time, lp_id
## dbl (2): pct_access, engagement_index
##
## 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.

colSums(is.na(engage2)) # check n/a

##             time            lp_id       pct_access engagement_index
##                0                1                0                0
```

As we learned from the first dataset, cleaning all n/a values in the pct_access and engagement_index columns can affect our analysis later. Therefore, from now on, we will only perform cleaning for the time and lp_id columns.

```
engage2 <- engage2 %>%
  mutate(time = replace(time, time == "1/1/2044", "1/1/2020"))

## While checking, we found a 'string' value, which poses a risk of
disrupting the data.
## Therefore, it is important to convert it to n/a before dropping it.
engage2 <- engage2 %>%
  mutate(time = replace(time, time == "not sure", NA)) %>%
  mutate(time = replace(time, time == " ", NA)) %>%
  filter(!is.na(lp_id)) %>%
  mutate(`lp_id` = as.numeric(`lp_id`)) # convert for merging

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

colSums(is.na(engage2)) # verify if adjustment is made

##             time            lp_id       pct_access engagement_index
##                0                1                0                0

# add new district_id column
engage2 <- mutate(engage2, district_id = 1039)
```

c. 1044.csv (Missouri). Similarly, we repeat the process for “1044.csv”.

```
engage3 = read_csv(file="Engagement Data/1044.csv") #rename for clarity

## Warning: One or more parsing issues, call `problems()` on your data frame
for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)

## Rows: 255722 Columns: 4
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): time
## dbl (3): lp_id, pct_access, engagement_index
##
## 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.

colSums(is.na(engage3))

##             time            lp_id       pct_access engagement_index
##                0                0                1            46128

engage3 <- engage3 %>%
  mutate(time = replace(time, time == "1/01/2050", "1/01/2020"))

# add new column
engage3 <- mutate(engage3, district_id = 1044)
```

d. 1052.csv (Illinois). Similarly, we repeat the observing process for “1052.csv”.

```
engage4 = read_csv(file="Engagement Data/1052.csv") #rename for clarity

## Rows: 91977 Columns: 4
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): time
## dbl (3): lp_id, pct_access, engagement_index
##
## 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.
```

```
# examine the dataset rows and columns
colSums(is.na(engage4))
```

```
##             time            lp_id       pct_access engagement_index
##                0                0                0            30762
```

```
engage4 <- engage4 %>%
  mutate(time = replace(time, time == "1/01/2033", "1/01/2020"))

# add new column
engage4 <- mutate(engage4, district_id = 1052)
```

Districts Data

We start by inspecting the dataset.

```
districts = read_csv(file="districts_info.csv")
```

```
## Rows: 233 Columns: 7
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr (6): state, locale, pct_black/hispanic, pct_free/reduced,
county_connect...
## dbl (1): district_id
##
## 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.
```

```
colSums(is.na(districts)) # check n/a
```

```
##              district_id                    state                   locale
##                        0                        0                        0
##       pct_black/hispanic         pct_free/reduced county_connections_ratio
##                        1                       28                       14
##             pp_total_raw
##                       58
```

```
districts <- districts %>% # rename columns for clarity
  rename(pct_black.hispanic = "pct_black/hispanic") %>%
  rename(pct_free.reduced = "pct_free/reduced")

# unique(districts$district_id) # inspect the data

  ## transform the data
districts <- districts %>%
  mutate(state = case_when(
    state %in% c("UTAH", "uTtah", "Utaah") ~ "Utah",
    state == "ConnectiCUT" ~ "Connecticut",
    state %in% c("NY City", "New Y0rk") ~ "New York",
    state == "Ohi0" ~ "Ohio",
    state == "District Of Columbia" ~ "Washington",
    state %in% c("whereabouts", "don\x92t know", "NaN") ~ NA,
    TRUE ~ state)) %>%
  filter(!is.na(state))

unique(districts$state) # verify if adjustment is made
```

```
##  [1] "Illinois"       "Utah"           "Wisconsin"      "North Carolina"
##  [5] "Missouri"       "Washington"     "Connecticut"    "Massachusetts"
##  [9] "New York"       "Indiana"        "Virginia"       "Ohio"
## [13] "New Jersey"     "California"     "Minnesota"      "Arizona"
## [17] "Texas"          "Tennessee"      "Florida"        "North Dakota"
## [21] "New Hampshire"  "Michigan"
```

```
  ## transform the data
districts <- districts %>%
  mutate(locale = case_when(
    locale %in% c("Sub", "C1ty") ~ "Suburb",
    locale == "Cit" ~ "City",
    TRUE ~ locale))

unique(districts$locale) # verify if adjustment is made
```

```
## [1] "Suburb" "City"   "Rural"  "Town"
```

```
  ## transform the data
districts <- districts %>%
  mutate(pct_black.hispanic = case_when(
    pct_black.hispanic == "NA" ~ NA,
    pct_black.hispanic == "[0, 0.2[" ~ "0-20%",
    pct_black.hispanic == "[0.2, 0.4[" ~ "20-40%",
    pct_black.hispanic == "[0.4, 0.6[" ~ "40-60%",
    pct_black.hispanic == "[0.6, 0.8[" ~ "60-80%",
    pct_black.hispanic == "[0.8, 1[" ~ "80-100%",
    TRUE ~ pct_black.hispanic)) %>%
  filter(!is.na(pct_black.hispanic))

unique(districts$pct_black.hispanic) # verify if adjustment is made
```

```
## [1] "0-20%"   "20-40%"  "40-60%"  "80-100%" "60-80%"
```

```
  ## transform the data
districts <- districts %>%
  mutate(pct_free.reduced = case_when(
    pct_free.reduced == "NA" ~ NA,
    pct_free.reduced == "[0, 0.2[" ~ "0-20%",
    pct_free.reduced == "[0.2, 0.4[" ~ "20-40%",
    pct_free.reduced == "[0.4, 0.6[" ~ "40-60%",
    pct_free.reduced == "[0.6, 0.8[" ~ "60-80%",
    pct_free.reduced == "[0.8, 1[" ~ "80-100%",
    TRUE ~ pct_free.reduced)) %>%
  filter(!is.na(pct_free.reduced))

unique(districts$pct_free.reduced) # verify if adjustment is made
```

```
## [1] "0-20%"   "20-40%"  "40-60%"  "60-80%"  "80-100%"
```

```
  ## transform the data
districts <- districts %>%
  mutate(county_connections_ratio = case_when(
    county_connections_ratio == "[0.18, 1[" ~ "<1",
    county_connections_ratio == "[1, 2[" ~ ">1",
    county_connections_ratio == "NA" ~ "NaN",
    TRUE ~ county_connections_ratio))

unique(districts$county_connections_ratio) # verify if adjustment is made
```

```
## [1] "<1" NA   ">1"
```

```
  ## transform the data
districts <- districts %>%
  mutate(
    pp_total_raw = case_when(
      pp_total_raw == "NA" ~ "NaN",
      pp_total_raw == "[4000, 6000[" ~ "4-6",
      pp_total_raw == "[6000, 8000[" ~ "6-8",
      pp_total_raw == "[8000, 10000[" ~ "8-10",
      pp_total_raw == "[10000, 12000[" ~ "10-12",
      pp_total_raw == "[12000, 14000[" ~ "12-14",
      pp_total_raw == "[14000, 16000[" ~ "14-16",
      pp_total_raw == "[16000, 18000[" ~ "16-18",
      pp_total_raw == "[18000, 20000[" ~ "18-20",
      pp_total_raw == "[20000, 22000[" ~ "20-22",
      pp_total_raw == "[22000, 24000[" ~ "22-24",
      pp_total_raw == "[32000, 34000[" ~ "32-34",
      TRUE ~ pp_total_raw))

unique(districts$pp_total_raw) # verify if adjustment is made
```

```
##  [1] "14-16" "6-8"   "10-12" "8-10"  "12-14" NA      "20-22" "18-20" "22-
24"
## [10] "16-18" "4-6"   "32-34"
```

Products Data

```
products = read_csv(file="products_info.csv")
```

```
## Rows: 372 Columns: 6
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr (5): URL, Product Name, Provider/Company Name, Sector(s), Primary
Essent...
## dbl (1): LP ID
##
## 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.
```

```
colSums(is.na(products))
```

```
##                      LP ID                        URL
##                          0                          0
##               Product Name      Provider/Company Name
##                          0                          1
##                  Sector(s) Primary Essential Function
##                         18                         20
```

```
products <- products %>% # rename columns
  rename(product_name = "Product Name") %>%
  rename(provider.company_name = "Provider/Company Name") %>%
  rename(sector = "Sector(s)") %>%
  rename(lp_id = "LP ID") %>%
  rename(primary_essential_function = "Primary Essential Function")

products <- products %>%
  filter(!is.na(product_name)) #dropping n/a

products <- products %>%
  mutate(sector = case_when(
    sector %in% c("PreK-122", "PreK-112", "PPreK-12", "pre kindergarten to yr
12",
                  "pre kindergarten to year 12") ~ "PreK-12",
    sector == "PreK-12; Higher; Corporate" ~ "PreK-12; Higher Ed; Corporate",
    sector == "not sure" ~ NA,
    TRUE ~ sector))

unique(products$sector)
```

```
## [1] "PreK-12"                       "PreK-12; Higher Ed"
## [3] "PreK-12; Higher Ed; Corporate" "Corporate"
## [5] NA                              "Higher Ed; Corporate"
```

```
# Extract function part
products$function_type <- sub("\\s*-.*", "",
products$primary_essential_function)

  ## transform the data
products <- products %>%
  mutate(function_type = replace(function_type, function_type == "CL", "LC"))
unique(products$function_type) #verify if adjustment is made
```

```
## [1] "LC"        "CM"        "SDO"       "LC/CM/SDO" NA
```

```
# Extract sub-function part
products$function_subtype <- sub(".*-\\s*", "",
products$primary_essential_function)
```

1.2 Wrangling

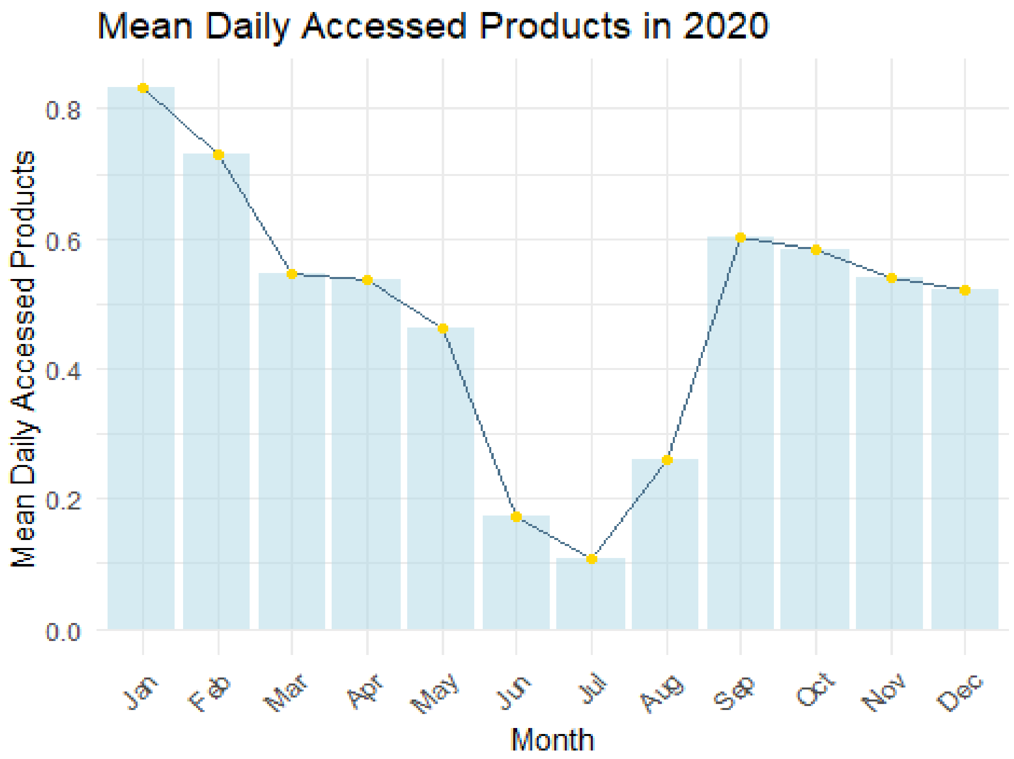
```
engagement <- bind_rows(engage1, engage2, engage3, engage4)

engagement <- engagement %>%
  left_join(districts, by = "district_id") # merge engagement with districts

engagement <- engagement %>%
  left_join(products, by = "lp_id") # merge engagement with products
```

2. Data Visualisation

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Warning: package 'viridis' was built under R version 4.3.3

## Loading required package: viridisLite
```

Section 1: Engagement During 2020

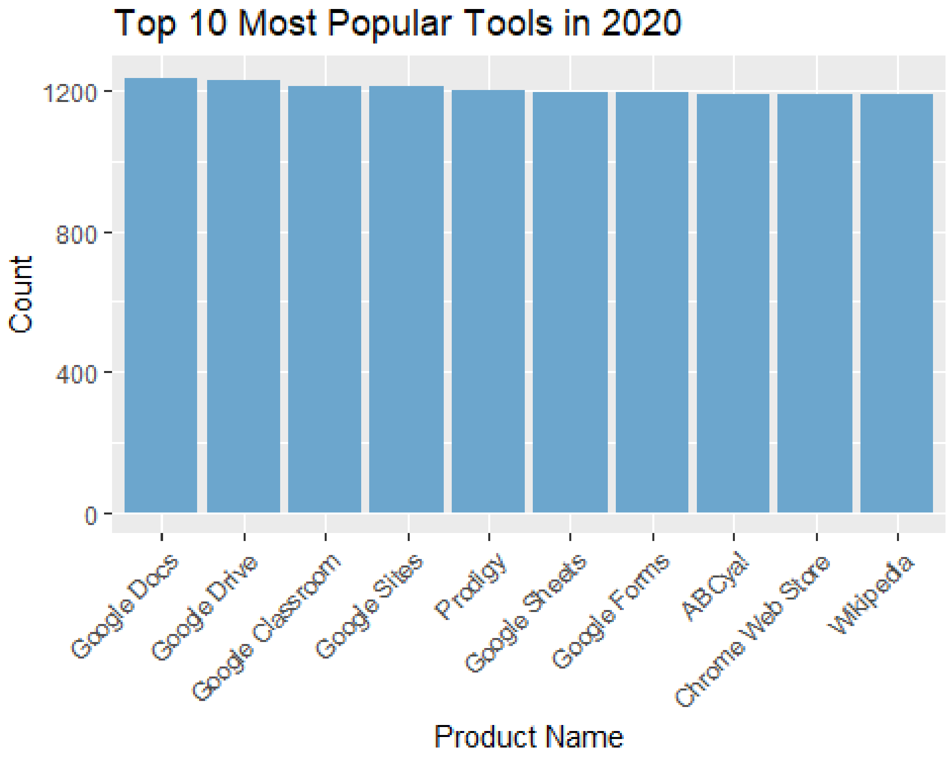
To explore the trend in 2020, a bar-line graph has been utilized. This combined graph format provides a comprehensive visualization by integrating two types of data representations: bars and lines. The bars represent the daily means of engagement metrics, offering a clear overview of the average engagement levels across different months. Meanwhile, the line graph outlines trends over time, allowing for a deeper analysis of the engagement patterns, especially regarding the pandemic outbreak or the school break.



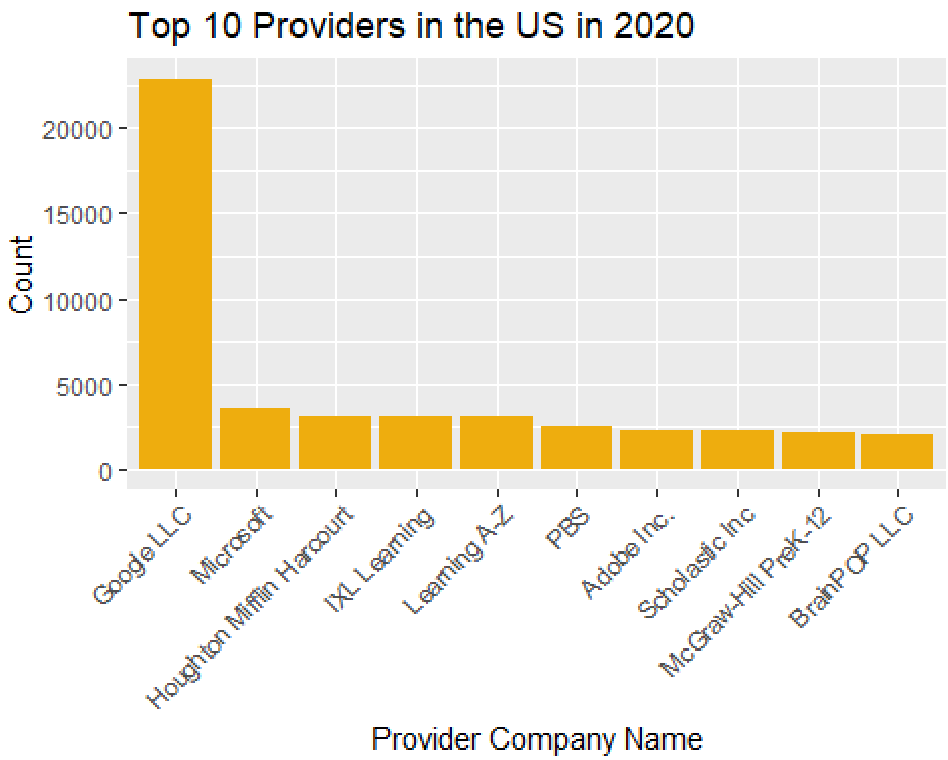
Section 2: Learning Platforms

2.1. Top Tools

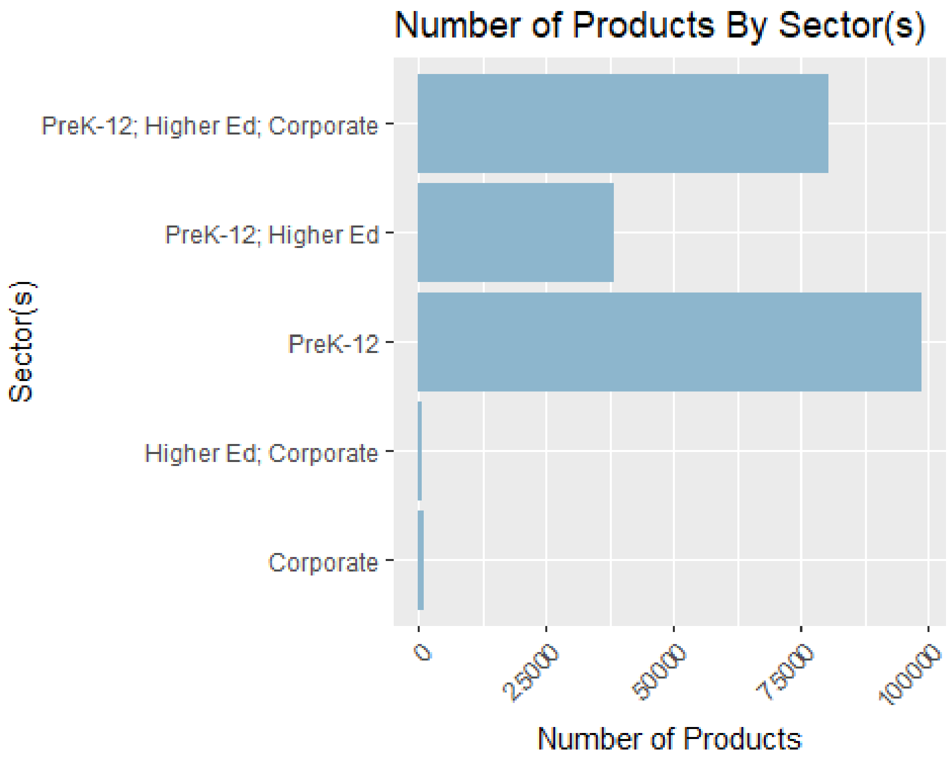
Regarding categorical values, a bar graph is considered the most effective as it allows for better comparison between different categories.



2.2. Top Providers



2.3. Top Sectors

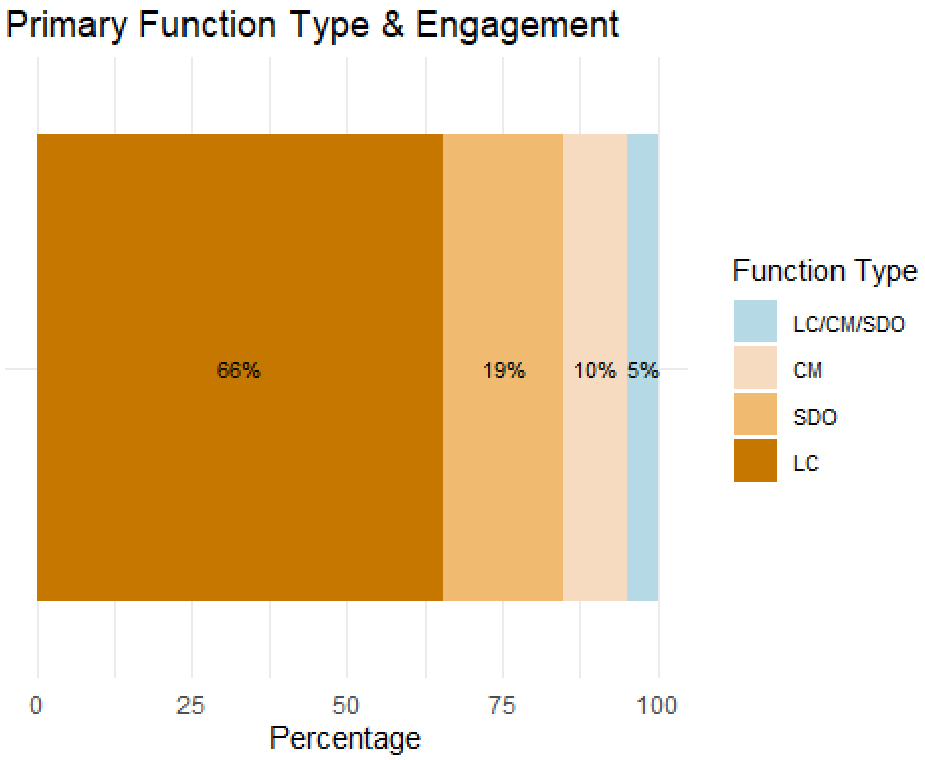


Section 3: Product Function and Engagement

In the context of examining the distribution of three primary function types (Learning & Curriculum, Classroom Management, and School & District Operations), a stacked bar graph is likely to be the most suitable. Although a pie chart was initially considered, it has been found (Hunt, 2019) that pie charts may not be optimal due to several reasons: (1) it is challenging for the audience to estimate accurately; (2) they complicate the matching of labels to slices; and (3) they may not effectively visualize small percentages.

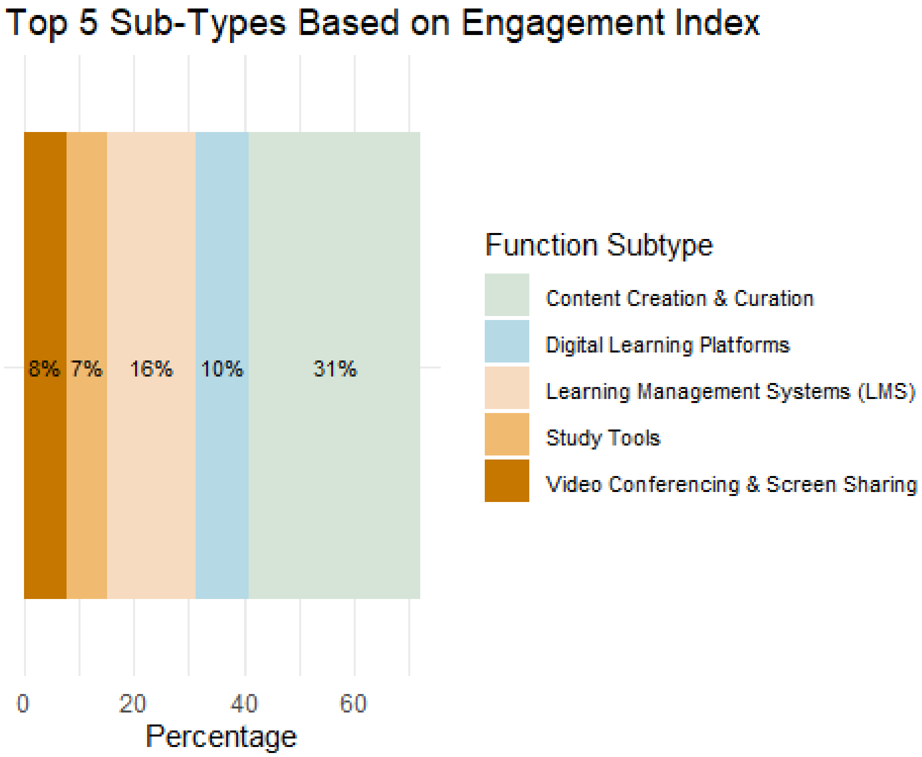
Meanwhile, stacked bar graphs offer a more intuitive way to communicate data, as they allow for a direct comparison of the quantities of each function type and sub-type within a single bar. Additionally, one application of a stacked bar graph is enabling readers to examine the rankings of items according to multiple attributes (Gratzl et al., 2013). Therefore, a stacked bar graph appears as the preferable choice.

3.1.



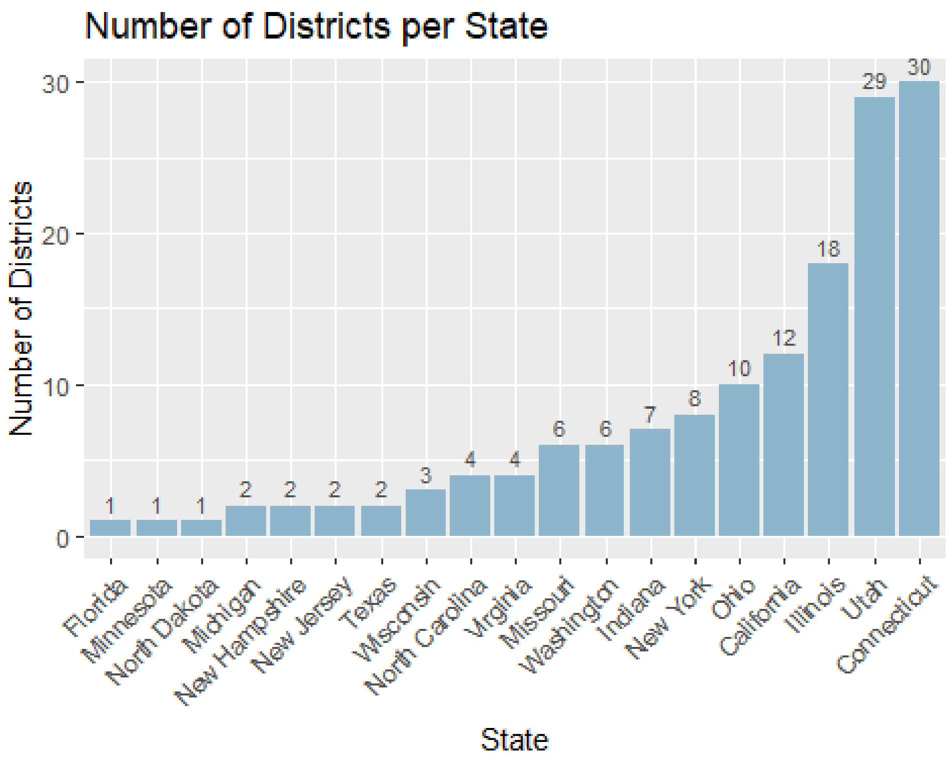
3.2.

```
## Selecting by Percentage
```

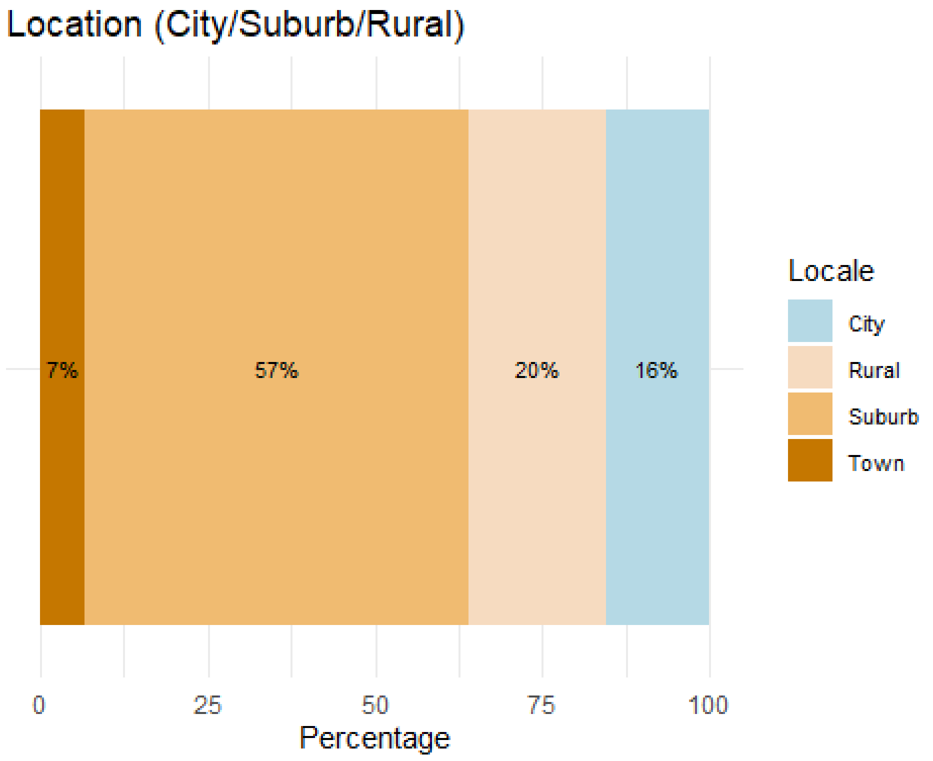


Section 4: Geographic Context

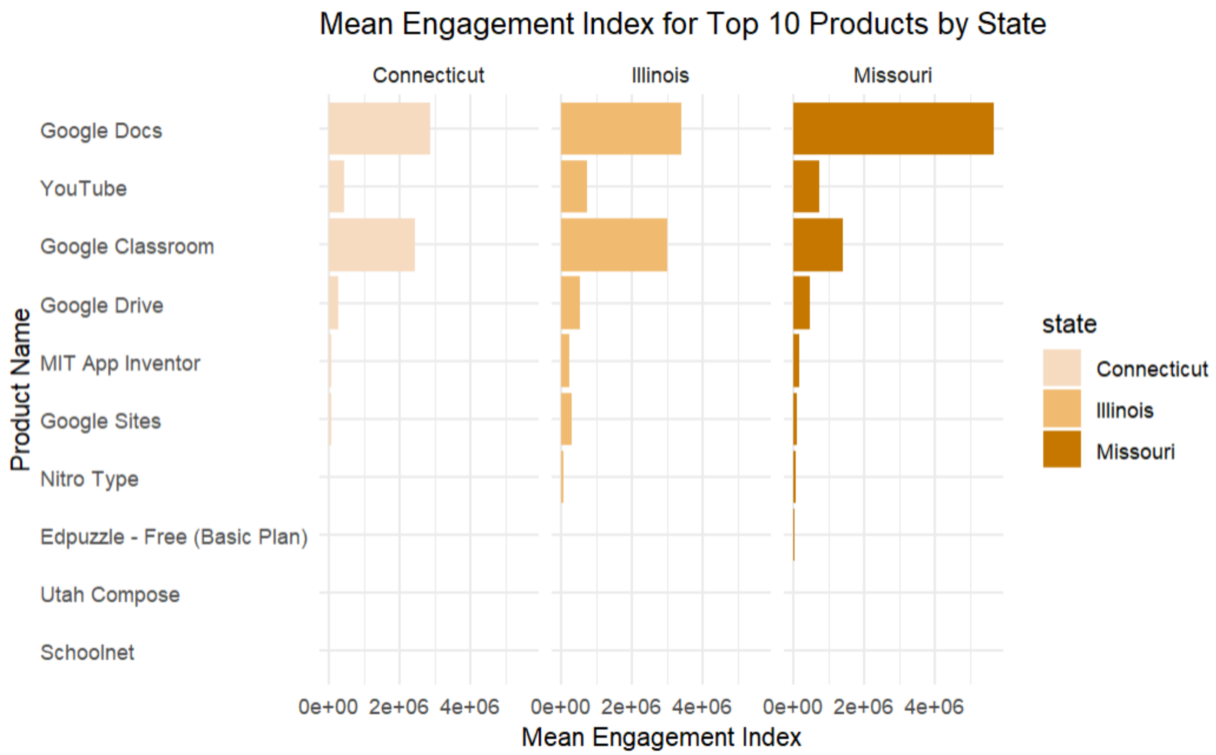
4.1. For the purpose of comparing the number of districts, bar graphs makes it straightforward to interpret and compare the data at a glance.



4.2. Similarly, as we need to clearly showcase the percentage of city/suburb/rural area, a stacked bar graph should be the best fit as it allows for a visual breakdown of each category's contribution to the whole.



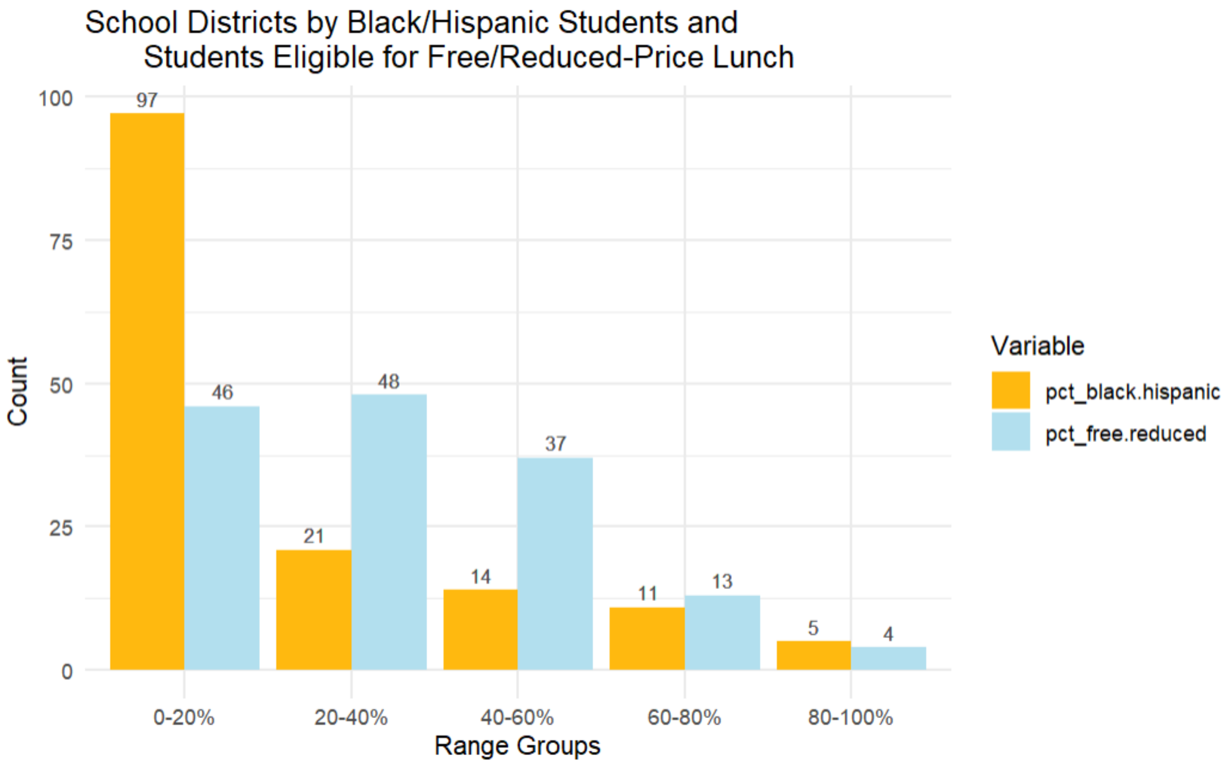
4.3.



Section 5: Demographic Context

5.1 Similarly, bar graphs are still efficient, especially since districts can be categorized by percentage range. Placing these graphs side by side helps viewers quickly identify patterns or discrepancies in the number of districts per range.

```
## [1] "0-20%"   "20-40%"  "40-60%"  "80-100%" "60-80%"

## [1] "0-20%"   "20-40%"  "40-60%"  "60-80%"  "80-100%"
```



We present a table for the audience to conveniently locate where the engagement datasets are in a bigger picture.

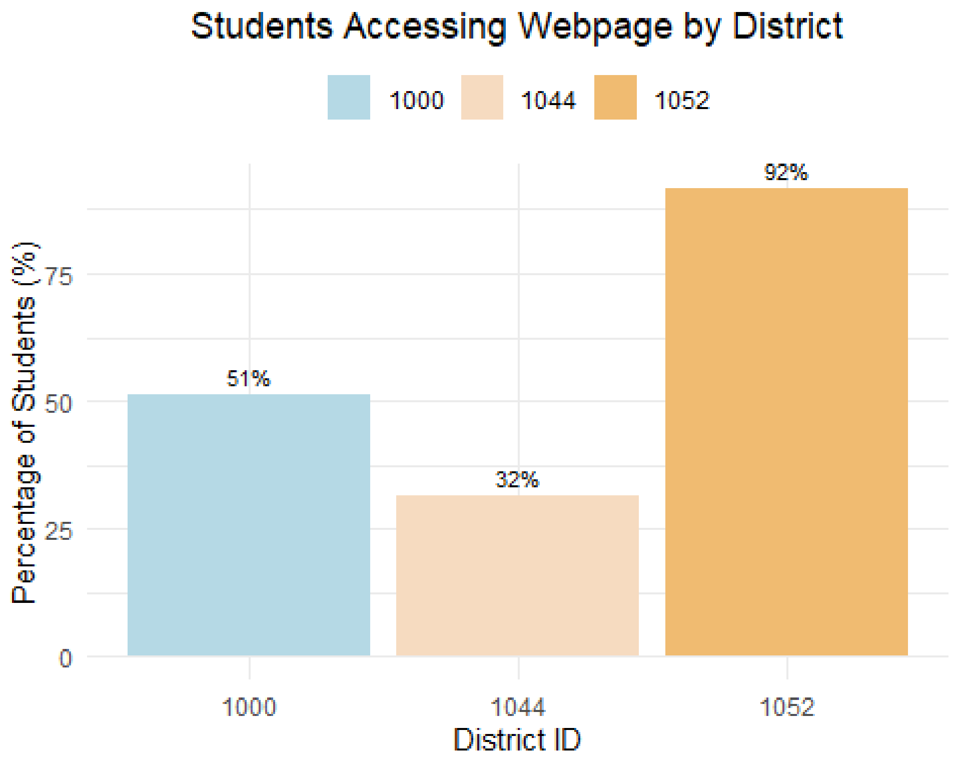
```
filtered_districts <- districts %>% #filter data
  filter(district_id %in% c(1000, 1044, 1052))
print(filtered_districts)
```

```
## # A tibble: 3 × 7
##   district_id state       locale pct_black.hispanic pct_free.reduced
##         <dbl> <chr>       <chr>  <chr>              <chr>
## 1        1044 Missouri    Suburb 0-20%              0-20%
## 2        1000 Connecticut Suburb 60-80%             20-40%
## 3        1052 Illinois    Suburb 20-40%             20-40%
## # ℹ 2 more variables: county_connections_ratio <chr>, pp_total_raw <chr>
```

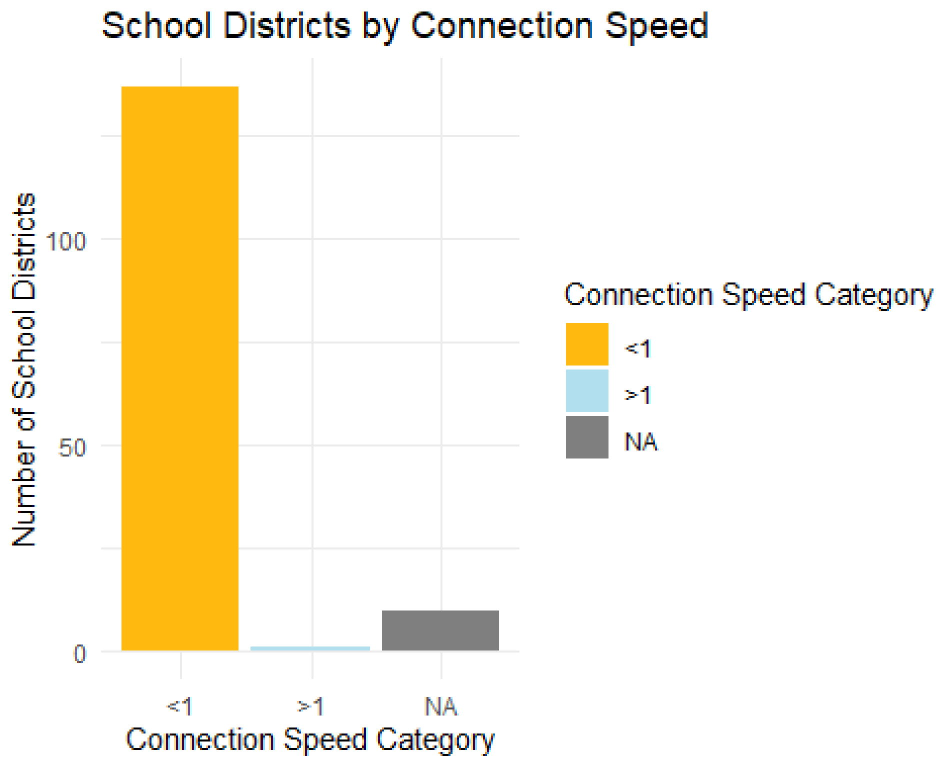
5.2 . Similarly, a bar graph can once again be utilized to effectively present this type of data. Given that the percentage of students is retrieved from ‘pct_access’, we calculate the mean for 2020 and compare them between the 3 districts:

- 1000: Connecticut (Suburb)
- 1044: Missouri (Suburb)
- 1052: Illinois (Suburb)

Due to limitations, we may consider each district as representative of its respective state.



5.3 . Similarly, as we have 3 categories for connection speed (">1", "<1", and "NA"), a bar graph will be an effective communication way to the audiences.



To take a deeper look at the outlier with a connection speed greater than 1, we retrieve the data into the following table.

```
# Districts with county_connections_ratio = ">1"
districts_greater_than_1 <- districts %>%
  filter(county_connections_ratio == ">1")

districts_greater_than_1

## # A tibble: 1 × 7
##   district_id state        locale pct_black.hispanic pct_free.reduced
##         <dbl> <chr>        <chr>  <chr>              <chr>
## 1        2872 North Dakota Rural  0-20%              0-20%
## # ℹ 2 more variables: county_connections_ratio <chr>, pp_total_raw <chr>
```

3. Findings & Recommendations

1. Trends of Student Engagement during 2020

The pandemic outbreak prompted a significant shift in educational practices, with educators globally transitioning to online platforms. Analysis of mean daily accessed products in 2020 reveals lower means in June, July, and August coinciding with typical school holidays, suggesting reduced academic engagement during vacation periods. This trend underscores the influence of external factors like school breaks on student online activity and highlights the adaptability of educational delivery methods during challenging times.

2. Learning Platforms

Among the top 10 popular tools, Google's suite including Docs, Drive, Classroom, and Sites holds a prominent position, alongside child-focused resources like ABCya! and Scholastic Inc., and essential platforms like Wikipedia and Dictionary.com. Notably, Google LLC leads the list as the top provider of the year, surpassing competitors in product accesses. Additionally, the analysis highlights PreK-12 as the dominant sector, emphasizing the prevalence of digital educational activities for younger learners. This finding underscores the prevalence of educational activities catering to younger learners within the digital landscape.

3. Primary Types & Sub-Types and Engagement Index

Among the three function types (Learning & Curriculum, Classroom Management, and School & District Operations), Learning & Curriculum dominates with 67% of the engagement index, exceeding the total of the other two. Content Creation & Curation, Learning Management Systems, and Digital Learning Platforms stand out as the most engaging sub-types, emphasizing the significance of educational content and platforms in driving student engagement, guiding educators and developers.

4. Geographic Context

From the graph 4.1, 4.2 and 4.3: - Connecticut, Utah, and Illinois have the highest numbers of districts. District 1000 is located in Connecticut, while District 1052 is from Illinois. Meanwhile, District 1044 is situated in Missouri, a state with a district count ranking in the middle of the list. - While most of the districts are located in suburb areas, all districts with given engagement dataset are also suburban. - There is a similarity regarding popular products for three districts of three different states. Thus, the geographical impact is not portraited clearly given this dataset. Overall, we observe a limitation as the investigation restricted around suburban areas and s small sample size, which hinder a comprehensive analysis.

5. Demographic Context

Illinois shows the highest average percentage of daily webpage access (92%), while District 1000 (Connecticut) and District 1044 (Missouri) exhibit notably lower rates at 51% and 32% respectively, despite all being suburban districts. While this trend aligns with Waller et al. (2020)'s argument that the pandemic exacerbated societal inequalities, further investigation is warranted. District 1052, with the highest access rate, has moderate percentages of Black/Hispanic students and students on free/reduced lunch, whereas District 1000's high percentages in these categories offer partial support to the argument. The dataset's bias towards districts with lower percentages of Black/Hispanic students and students on free/reduced lunch should be considered, alongside the limited focus on demographic factors like household income and type.

6. Recommendations

- 6.1/ **For educators.** They are suggested to take advantage of a variety of educational resources beyond traditional textbooks and be mindful of the varying access rates to online resources among districts, which may reflect socioeconomic disparities. Additionally, although the dataset may lack clear geographical impacts, leverage insights from demographic data to tailor educational approaches.
- 6.2/ **For digital providers**. They are recommended to prioritize accessibility features (screen reader compatibility, language support, and user-friendly interfaces to enhance accessibility for all learners) in digital learning platforms to ensure inclusivity for students with diverse needs. Secondly, they need to encourage collaboration between educators and platform developers to create innovative educational tools. Lastly, they are encouraged to take proactive approach to mitigate biases by collecting more comprehensive demographic data.

References

Gratzl, S., Lex, A., Gehlenborg, N., Pfister, H., & Streit, M. (2013). LineUp: Visual analysis of multiattribute rankings. *IEEE Transactions on Visualization and Computer Graphics, 19*(12), 2277–2286. https://doi.org/10.1109/TVCG.2013.173

Hunt, C. (2019, November 25). Why you shouldn't use pie charts. *Statistical Consulting Centre*. https://scc.ms.unimelb.edu.au/resources/data-visualisation-and-exploration/no_pie-charts

Stanistreet, P., Elfert, M., & Atchoarena, D. (2020). Education in the age of COVID-19: Understanding the consequences. *International Review of Education, 66*(5), 627–633. https://doi.org/10.1007/s11159-020-09880-9

Waller, R., Hodge, S., Holford, J., Milana, M., & Webb, S. (2020). Lifelong education, social inequality and the COVID-19 health pandemic. *International Journal of Lifelong Education, 39*(3), 243–246. https://doi.org/10.1080/02601370.2020.1790267