Data science community analysis

9/22/2021

Contents

---

1 Introduction

The job descriptions in data science community are diverse in terms of requirements with various titles. Moreover, evolving studies around data management have constantly evolving needs. Consequently new job profiles are opening up and its is not always straightforward to compare these roles. Significance of higher education is hard to analyze because indivudials claim to have achieved these roles through online courses/bootcamps. Through this project we strive to answer some of such questions that are relevant to our cohort as we prepare to enter the field as professionals in the next 2 years.

For our purpose we have considered the Kaggle's Annual Survey on Machine Learning and Data Science from 2017-20. The respondents are from various parts of the world and can be assumed to be a good representation of the broader community. For limiting the scope of the project, most of the analysis focuses on US respondents. Besides limiting scope it is more relevant to our cohort as well. However the study is generic and by reducing country of residence filters in feature modeling, we can extend the analysis to include international prospects. Next we discuss the specific questions we are going to analyze.

2 Problem Statement

The purpose of this project is to analyze Kaggle's data science community and answer the following questions:

1. How is the gender diversity in the community?
2. How does compensation trend for different job categories?
3. Is higher education a common element for different job categories?
4. How does coding experience effect the compensation in long run?
5. What programming languages are more relevant for different jobs?

As we mentioned before, our dataset consists of the *Kaggle Machine Learning & Data Science Survey* from 2017-20. Each year survey data contains more than 200 columns with several thousand respondents. Let us load and see the dimensions of the dataset which are hosted and accessed here from github for easy reproducibility.

```
base_url <- "https://raw.githubusercontent.com/sprihap/KaggleDatasets/master/"
survey_urls <- str_c(base_url, c("kaggle-survey-2017/multipleChoiceResponses.csv",
                                 "kaggle-survey-2018/multipleChoiceResponses.csv",
                                 "kaggle-survey-2019/multiple_choice_responses.csv",
                                 "kaggle-survey-2020/kaggle_survey_2020_responses.csv"))

data_2017 = read.csv(survey_urls[1], header = TRUE,
                     check.names = FALSE, stringsAsFactors = FALSE)
data_2018 = read.csv(survey_urls[2], header = TRUE,
                     check.names = FALSE, stringsAsFactors = FALSE)
data_2019 = read.csv(survey_urls[3], header = TRUE,
                     check.names = FALSE, stringsAsFactors = FALSE)
data_2020 = read.csv(survey_urls[4], header = TRUE,
                     check.names = FALSE, stringsAsFactors = FALSE)
```

The rows and columns of each year's datasets can be displayed through *nrow* and *ncol* function.

```
nrow(data_2017)
```

```
## [1] 16716
```

```
ncol(data_2017)
```

```
## [1] 228
```

```
nrow(data_2018)
```

```
## [1] 23860
```

```
ncol(data_2018)
```

```
## [1] 395
```

```
nrow(data_2019)
```

```
## [1] 19718
```

```
ncol(data_2019)
```

```
## [1] 246
```

```
nrow(data_2020)
```

```
## [1] 20037
```

```
ncol(data_2020)
```

```
## [1] 355
```

The data set is not consistent across years and needs careful analysis to clean and extract data relevant to our problem statement. Description of each column for years is available at our hosted datasets on github at *https://github.com/sprihap/KaggleDatasets*.

3 Feature modeling

Our first step is to analyze the dataset. As we saw the data has very high dimensions. We need to extract our relevant features in consistent schema and then draw visualizations to analyze.

We prepare two data frames:

1. For analyzing the demography parameter like gender diversity, we consider all the respondents.
2. For other questions we want to focus on US residents and USD incomes and hence will need to filter data.

3.1 Gender level feature model

To generate gender level feature model we identify the columns that map to that in each year and then mutate to get the percentage of gender identified in different years.

```
gender_2017 <- data_2017 %>% rename(gender = GenderSelect) %>%
  group_by(gender) %>% summarize(Y2017 = n()) %>% ungroup() %>%
  mutate(Y2017 = (Y2017 * 100) / sum(Y2017))
gender_2018 <- data_2018[-c(1),] %>% rename(gender = Q1) %>%
  group_by(gender) %>% summarize(Y2018 = n()) %>% ungroup()%>%
  mutate(Y2018 = (Y2018 * 100) / sum(Y2018))
gender_2019 <- data_2019[-c(1),] %>% rename(gender = Q2) %>%
  group_by(gender) %>% summarize(Y2019 = n()) %>% ungroup()%>%
  mutate(Y2019 = (Y2019 * 100) / sum(Y2019))
gender_2020 <- data_2020[-c(1),] %>% rename(gender = Q2) %>%
  group_by(gender) %>% summarize(Y2020 = n()) %>% ungroup()%>%
  mutate(Y2020 = (Y2020 * 100) / sum(Y2020))

# Make some renamings to map to same values
gender_2017 <- gender_2017 %>%
  mutate(gender = case_when(
    gender == "Non-binary, genderqueer, or gender non-conforming" ~ "Nonbinary",
    gender == "A different identity" ~ "Prefer to self-describe",
    gender == "" ~ "Prefer not to say",
    TRUE ~ gender
  ))

gender_2020 <- gender_2020 %>%
  mutate_at("gender", str_replace, "Man", "Male") %>%
  mutate_at("gender", str_replace, "Woman", "Female")

gender <- gender_2017 %>%
  full_join(gender_2018) %>%
  full_join(gender_2019) %>%
  full_join(gender_2020)
```

```
## Joining, by = "gender"
## Joining, by = "gender"
## Joining, by = "gender"
```

```
gender <- gender %>%
  pivot_longer(cols = c(Y2017, Y2018, Y2019, Y2020),
               names_to = "year",
               values_to = "percent") %>%
  mutate(year = factor(year), gender = factor(gender))

glimpse(gender)
```

```
## Rows: 20
## Columns: 3
## $ gender  <fct> Prefer not to say, Prefer not to say, Prefer not to say, Prefe~
## $ year    <fct> Y2017, Y2018, Y2019, Y2020, Y2017, Y2018, Y2019, Y2020, Y2017,~
## $ percent <dbl> 0.5683178, 1.4250388, 1.6128214, 1.3126373, 0.9511845, 0.33111~
```

Now that we have gender level feature data, we will generate US residents focused features in next section.

3.2 Compensation and other parameters feature model for US residents

For modeling compensation in relation to other parameters, we first try to extract the compensation of different roles in each year. This needs a significant effort on understanding the data and mapping the columns form each year to a consistent schema that we can read and analyze. As proposed before, we are going to focus on people who report their earnings in USD (2017 Survey) or are living in US (2018-20 Surveys).

This feature extraction introduces the following challenges that we tackle through cleanup:

1. The surveys across years aren't consistent in terms of schema. Each year columns change for different attributes.
2. The expected responses for similar questions are also different and need different processing / filtering strategies.
3. Compensation in 2017 was collected as an absolute number but in later years was converted to a bucket selection.
4. The job profiles are diverse across years that respondents can chose.
5. The coding experience is also recorded as an interval and isn't directly translated to real line.

To handles these we carefully read/extract each year's relevant columns and try different techniques to reach a consistent data for jobs and compensations. We use the average of closed buckets as expected compensation and coding duration to map the intervals on real line. For buckets with no upper limit we assume 1.5 times the upper value for average for compensation.

For comparing different jobs we reduce the options by generating categories for similar job profiles. We will consider the following categories with relevance to job seekers:

1. Data/Business Analyst (labeled *Analyst* in code)
2. Data Scientist
3. Machine Learning Engineer (labeled *ML Engineer* in code)
4. Software Engineer (labeled *SW Engineer* in code)
5. Others (which includes consultants, students, statistician etc.)

A demerit of *Others* bucket is that those job categories are not significant represented individually in Kaggle's survey universe. For example, there are less than 500 statisticians and research scientists combined even though they play a significant role in evolution of field.

We also needed to cleanup and map education level by their degree name to consistent strings. The original values in data contain special characters which need to be transformed for comparison and analysis.

Next we extract the desired features as tibble for each year below using techniques described.

3.2.1 2017

We filter the data by USD and valid compensation amounts. To find unwanted values we manually bucketed the currencies to count distinct values and filtered invalid values out.

```
comp_2017 <- data_2017 %>%
  filter((CompensationCurrency == "USD") &
           (!is.na(CompensationAmount)) &
           (CompensationAmount != "") &
           (CompensationAmount != "-") &
           (str_length(CurrentJobTitleSelect) > 0)) %>%
  mutate(Compensation = as.numeric(str_replace_all(CompensationAmount, ",", "")),
         Degree = FormalEducation,
         Major = MajorSelect,
         Job = CurrentJobTitleSelect,
         CodingDuration = Tenure,
         JobCategory = case_when(
           Job %in% c("Business Analyst", "Data Analyst") ~ "Analyst",
           Job == "Data Scientist" ~ "Data Scientist",
           Job == "Machine Learning Engineer" ~ "ML Engineer",
           Job == "Software Developer/Software Engineer" ~ "SW Engineer",
           TRUE ~ "Other"
         ),
         EducationLevel = case_when(
           str_detect(Degree, "Bachelor.*") ~ "Bachelors",
           str_detect(Degree, "Master.*") ~ "Masters",
           Degree == "Doctoral degree" ~ "PhD",
           str_detect(Degree, "I did not complete.*") ~ "Unfinished college",
           str_detect(
             Degree, "Some college/university study without.*") ~ "Unfinished college",
           Degree == "Professional degree" ~ "Professional Degree",
           Degree == "I prefer not to answer" ~ "Unknown",
           TRUE ~ Degree
         )) %>%
  select(Job, JobCategory, Degree, EducationLevel, Major,
         CodingDuration, Compensation) %>%
  filter(Compensation > 0)

summary(comp_2017)
```

```
##      Job             JobCategory           Degree          EducationLevel
##  Length:1533        Length:1533        Length:1533        Length:1533
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##     Major           CodingDuration      Compensation
##  Length:1533        Length:1533        Min.   :      1
```

```
##  Class :character   Class :character   1st Qu.:  60000
##  Mode  :character   Mode  :character   Median :  96000
##                                        Mean   : 115587
##                                        3rd Qu.: 140000
##                                        Max.   :9999999
```

At this stage, we see above some outliers of reported income in USD of 9,999,999 and 2,500,000 (possibly errors of currency conversion), which needs to be cleaned up.

```
comp_2017 <- comp_2017 %>% filter(Compensation < 2000000)
```

3.2.2 2018

The schema of 2018 changes significantly. Users have a choice of selecting their country of residence and compensation is an interval now. Possible response options to questions like job profile, education also change. To map compensation from interval to real line we split and use the average of bucket as likely value. As proposed earlier, for unbounded intervals we compute 1.5 times the lower bound.

```
comp_2018 <- data_2018[-c(1),] %>%
  filter((Q9 != "") &
           (Q9 != "I do not wish to disclose my approximate yearly compensation") &
           (Q3 == "United States of America")) %>%
  mutate(Job = Q6,
         Degree = Q4,
         Major = Q5,
         Q9 = str_replace_all(Q9, ",|\\+", ""), # Filter the characters to split
         CompensationRange = Q9,
         CodingDuration = Q24,
         JobCategory = case_when(
           Job %in% c("Business Analyst", "Data Analyst") ~ "Analyst",
           Job == "Data Scientist" ~ "Data Scientist",
           Job == "Machine Learning Engineer" ~ "ML Engineer",
           Job == "Software Engineer" ~ "SW Engineer",
           TRUE ~ "Other"
         ),
         EducationLevel = case_when(
           str_detect(Degree, "Bachelor.*") ~ "Bachelors",
           str_detect(Degree, "Master.*") ~ "Masters",
           Degree == "Doctoral degree" ~ "PhD",
           Degree == "No formal education past high school" ~ "Unfinished college" ,
           str_detect(
             Degree, "Some college/university study without.*") ~ "Unfinished college",
           Degree == "Professional degree" ~ "Professional Degree",
           Degree == "I prefer not to answer" ~ "Unknown",
           TRUE ~ Degree
         )) %>%
  separate(Q9, c("MinComp", "MaxComp"), "-") %>%
  select(Job, JobCategory, Degree, EducationLevel, Major, CodingDuration,
         CompensationRange, MinComp, MaxComp) %>%
  mutate(MinComp = as.numeric(MinComp),
         MaxComp = as.numeric(MaxComp),
         AvgComp = case_when(
           is.na(MaxComp) ~ 1.5 * MinComp,
```

```
          TRUE ~ (MinComp + MaxComp)/2
        ))
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 22 rows [341,
## 456, 665, 914, 1314, 1430, 1988, 2306, 2385, 2532, 2542, 2642, 2686, 2722, 2727,
## 2747, 2759, 2825, 2898, 2926, ...].
```

```
summary(comp_2018)
```

```
##      Job             JobCategory           Degree          EducationLevel
##  Length:3393        Length:3393        Length:3393        Length:3393
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##     Major           CodingDuration     CompensationRange     MinComp
##  Length:3393        Length:3393        Length:3393        Min.   :     0
##  Class :character   Class :character   Class :character   1st Qu.:    40
##  Mode  :character   Mode  :character   Mode  :character   Median :    80
##                                                           Mean   :  3327
##                                                           3rd Qu.:   125
##                                                           Max.   :500000
##
##     MaxComp          AvgComp
##  Min.   : 10000   Min.   :  5000
##  1st Qu.: 50000   1st Qu.: 25020
##  Median : 90000   Median : 45040
##  Mean   :106141   Mean   : 57632
##  3rd Qu.:150000   3rd Qu.: 75062
##  Max.   :500000   Max.   :750000
##  NA's   :22
```

3.2.3 2019

The variation of schema from 2018 to 2019 is not as bad but needs different values for buckets to income. There was some rewording but mostly same analysis is repeated as we did for 2018.

```
comp_2019 <- data_2019[-c(1),] %>%
  filter((Q10 != "") &
           (Q10 != "I do not wish to disclose my approximate yearly compensation") &
           (Q3 == "United States of America")) %>%
  mutate(Job = Q5,
         Degree = Q4,
         Q10 = str_replace_all(Q10, "\\$|>|,|\\+|( )+", ""), # Filter again to split
         CompensationRange = Q10,
         CodingDuration = Q15,
         JobCategory = case_when(
           Job %in% c("Business Analyst", "Data Analyst") ~ "Analyst",
           Job == "Data Scientist" ~ "Data Scientist",
           Job == "Machine Learning Engineer" ~ "ML Engineer",
           Job == "Software Engineer" ~ "SW Engineer",
           TRUE ~ "Other"
```

```
        ),
        EducationLevel = case_when(
          str_detect(Degree, "Bachelor.*") ~ "Bachelors",
          str_detect(Degree, "Master.*") ~ "Masters",
          Degree == "Doctoral degree" ~ "PhD",
          Degree == "No formal education past high school" ~ "Unfinished college" ,
          str_detect(
            Degree, "Some college/university study without.*") ~ "Unfinished college",
          Degree == "Professional degree" ~ "Professional Degree",
          Degree == "I prefer not to answer" ~ "Unknown",
          TRUE ~ Degree
        )) %>%
  separate(Q10, c("MinComp", "MaxComp"), "-") %>%
  select(Job, JobCategory, Degree, EducationLevel, CodingDuration,
         CompensationRange, MinComp, MaxComp) %>%
  mutate(MinComp = as.numeric(MinComp),
         MaxComp = as.numeric(MaxComp),
         AvgComp = case_when(
           is.na(MaxComp) ~ 1.5 * MinComp,
           TRUE ~ (MinComp + MaxComp)/2
         ))
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 26 rows [44,
## 130, 132, 267, 309, 516, 525, 564, 633, 712, 764, 769, 815, 825, 882, 970, 1058,
## 1086, 1342, 1557, ...].
```

```
summary(comp_2019)
```

```
##      Job             JobCategory           Degree          EducationLevel
##  Length:2134        Length:2134        Length:2134        Length:2134
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  CodingDuration     CompensationRange     MinComp          MaxComp
##  Length:2134        Length:2134        Min.   :     0   Min.   :   999
##  Class :character   Class :character   1st Qu.: 70000   1st Qu.: 79999
##  Mode  :character   Mode  :character   Median :100000   Median :124999
##                                        Mean   :111777   Mean   :135318
##                                        3rd Qu.:150000   3rd Qu.:199999
##                                        Max.   :500000   Max.   :500000
##                                                         NA's   :26
##     AvgComp
##  Min.   :   499.5
##  1st Qu.: 74999.5
##  Median :112499.5
##  Mean   :128814.7
##  3rd Qu.:174999.5
##  Max.   :750000.0
##
```

3.2.4 2020

Schema still remains similar to 2019 and a similar study is repeated. However, for 2020 we also will extract programming language trends so that we can answer part 5 of our problem statement. These columns are such that each person can select and provide feedback for multiple languages and we use their responses as an indicator of relevance in their job profile.

```
comp_2020 <- data_2020[-c(1),] %>%
  filter((Q24 != "") &
           (Q24 != "I do not wish to disclose my approximate yearly compensation") &
           (Q3 == "United States of America")) %>%
  mutate(Job = Q5,
         Degree = Q4,
         Q24 = str_replace_all(Q24, "\\$|>|,|\\+|( )+", ""),
         CompensationRange = Q24,
         CodingDuration = Q6,
         JobCategory = case_when(
           Job %in% c("Business Analyst", "Data Analyst") ~ "Analyst",
           Job == "Data Scientist" ~ "Data Scientist",
           Job == "Machine Learning Engineer" ~ "ML Engineer",
           Job == "Software Engineer" ~ "SW Engineer",
           TRUE ~ "Other"
         ),
         EducationLevel = case_when(
           str_detect(Degree, "Bachelor.*") ~ "Bachelors",
           str_detect(Degree, "Master.*") ~ "Masters",
           Degree == "Doctoral degree" ~ "PhD",
           Degree == "No formal education past high school" ~ "Unfinished college" ,
           str_detect(
             Degree, "Some college/university study without.*") ~ "Unfinished college",
           Degree == "Professional degree" ~ "Professional Degree",
           Degree == "I prefer not to answer" ~ "Unknown",
           TRUE ~ Degree
         ),
         Python = na_if(Q7_Part_1,""),
         R = na_if(Q7_Part_2,""),
         Sql = na_if(Q7_Part_3,""),
         C = na_if(Q7_Part_4,""),
         Cpp = na_if(Q7_Part_5,""),
         Java = na_if(Q7_Part_6,""),
         Javascript = na_if(Q7_Part_7,""),
         Julia = na_if(Q7_Part_8,""),
         Swift = na_if(Q7_Part_9,""),
         Bash = na_if(Q7_Part_10,""),
         MATLAB = na_if(Q7_Part_11,""),
         NoLanguage = na_if(Q7_Part_12,"")) %>%
  separate(Q24, c("MinComp", "MaxComp"), "-") %>%
  select(Job, JobCategory, Degree, EducationLevel, CodingDuration,
         CompensationRange, MinComp, MaxComp, Python, R, Sql, C, Cpp, Java,
         Javascript, Julia, Swift, Bash, MATLAB, NoLanguage) %>%
  mutate(MinComp = as.numeric(MinComp),
         MaxComp = as.numeric(MaxComp),
         AvgComp = case_when(
           is.na(MaxComp) ~ 1.5 * MinComp,
```

```
          TRUE ~ (MinComp + MaxComp)/2
        ))
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 12 rows [109,
## 141, 240, 852, 855, 926, 952, 1064, 1122, 1131, 1259, 1484].
```

```
summary(comp_2020)
```

```
##      Job             JobCategory           Degree          EducationLevel
##  Length:1484        Length:1484        Length:1484        Length:1484
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  CodingDuration     CompensationRange     MinComp          MaxComp
##  Length:1484        Length:1484        Min.   :     0   Min.   :   999
##  Class :character   Class :character   1st Qu.: 70000   1st Qu.: 79999
##  Mode  :character   Mode  :character   Median :100000   Median :124999
##                                        Mean   :106521   Mean   :131139
##                                        3rd Qu.:150000   3rd Qu.:199999
##                                        Max.   :500000   Max.   :500000
##                                                         NA's   :12
##     Python               R                Sql                C
##  Length:1484        Length:1484        Length:1484        Length:1484
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      Cpp                Java            Javascript           Julia
##  Length:1484        Length:1484        Length:1484        Length:1484
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##     Swift               Bash              MATLAB           NoLanguage
##  Length:1484        Length:1484        Length:1484        Length:1484
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##     AvgComp
##  Min.   :   499.5
##  1st Qu.: 74999.5
##  Median :112499.5
##  Mean   :122343.2
##  3rd Qu.:174999.5
```

```
##  Max.   :750000.0
##
```

4 Analysis

Now that we have relevant data extracted in our feature models as tibbles, we can proceed to our analysis and discoveries.

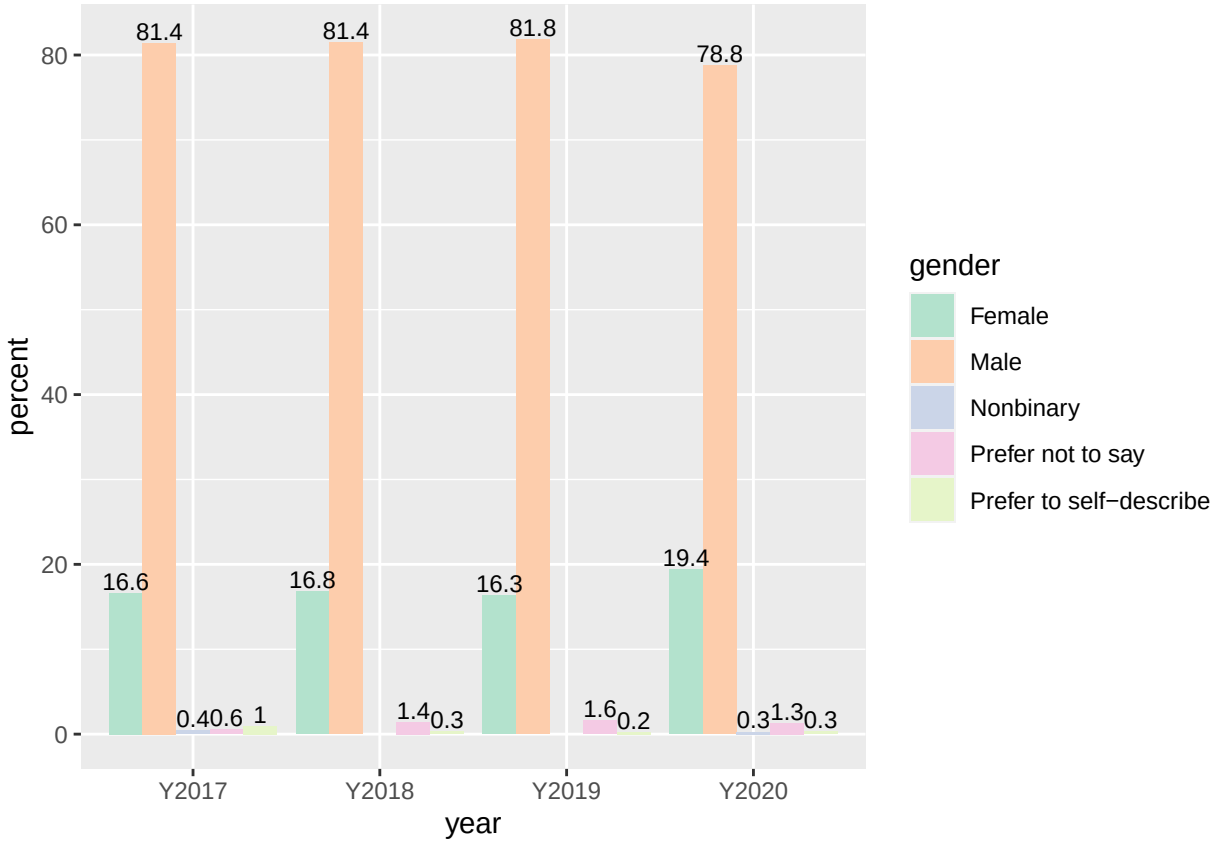
4.1 How is the gender diversity in the community?

To analyze the gender diversity we equate the percentage of participants as proportional to gender in the community. We will use the feature model extracted before. Below we plot the the histogram of the distribution and compare the percentage of males across the years.

```
ggplot(data=gender, aes(x = year, y = percent, fill = gender)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_fill_brewer(palette = "Pastel2") +
  geom_text(aes(label = round(percent, 1)), vjust = -0.2, size = 3,
            position = position_dodge(0.9))
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```

```
## Warning: Removed 2 rows containing missing values (geom_text).
```



From the graph above we see the following:

1. The industry has large percentage of individuals who identify as males, even though it has slightly reduced in size to ~79% from 2019 to 2020.

2. Females have increased in percentage by ~3% from 2019 to 2020 but still have very low representation.
3. Nonbinary gender and those who don't want to identify is very small and less than 2% in 2020.

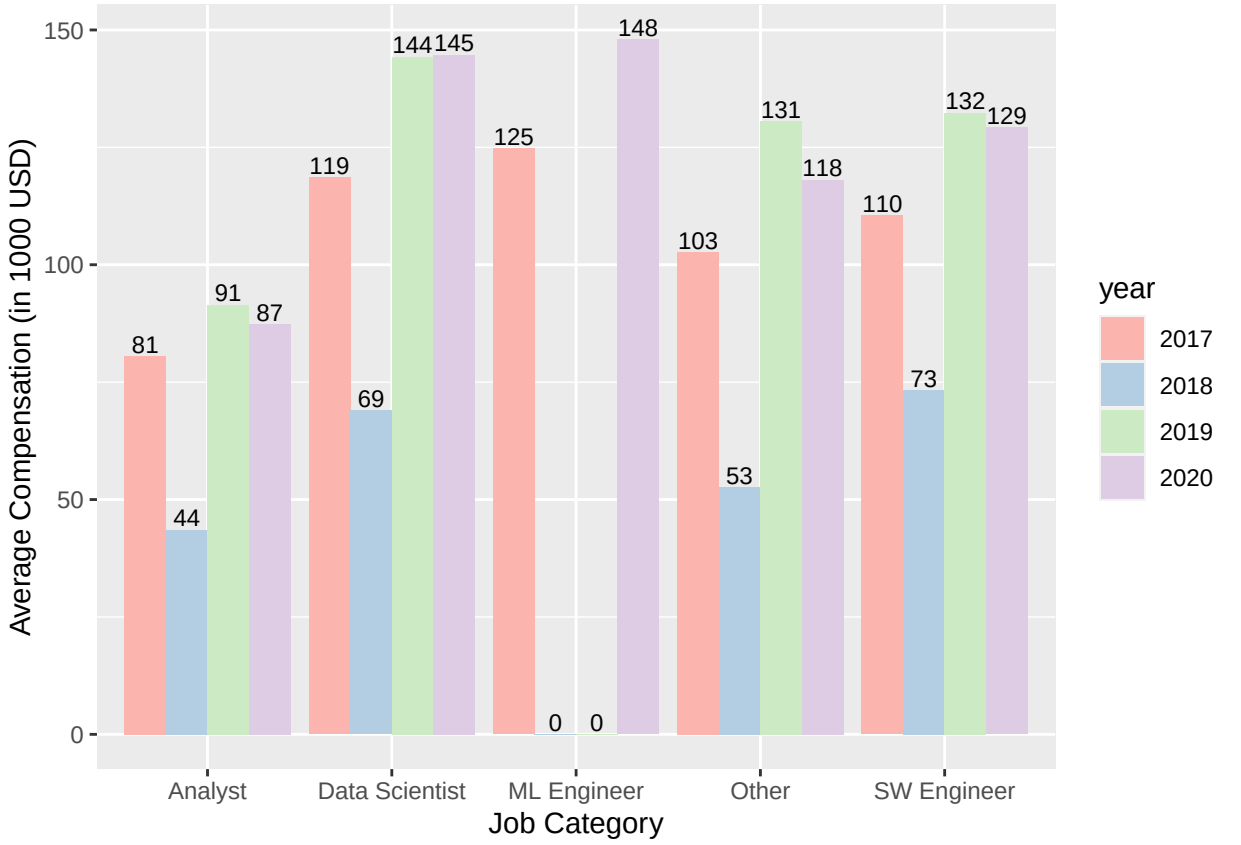
These observations show the community has large room for growth in diversification.

4.2 Compensation changes across the years

To compare compensations we focus on compensation by job and filter out statistics by grouping on it

```
comp_by_job_2017 <- comp_2017 %>%
  group_by(JobCategory) %>%
  summarize(year = "2017",
            count = n(),
            avgComp = mean(Compensation),
            minComp = min(Compensation),
            maxComp = max(Compensation))

comp_by_job_2018 <- comp_2018 %>%
  group_by(JobCategory) %>%
  summarize(year = "2018",
            count = n(),
            avgComp = mean(AvgComp),
            minComp = min(AvgComp),
            maxComp = max(AvgComp))

# Add missing ML Engineer category
comp_by_job_2018 <- comp_by_job_2018 %>%
  add_row(tibble_row(year = "2018",
                     JobCategory = "ML Engineer",
                     count = 0,
                     avgComp = 0,
                     minComp = 0,
                     maxComp = 0))

comp_by_job_2019 <- comp_2019 %>%
  group_by(JobCategory) %>%
  summarize(year = "2019",
            count = n(),
            avgComp = mean(AvgComp),
            minComp = min(AvgComp),
            maxComp = max(AvgComp))

# Add missing ML Engineer category
comp_by_job_2019 <- comp_by_job_2019 %>%
  add_row(tibble_row(year = "2019",
                     JobCategory = "ML Engineer",
                     count = 0,
                     avgComp = 0,
                     minComp = 0,
                     maxComp = 0))

comp_by_job_2020 <- comp_2020 %>%
  group_by(JobCategory) %>%
  summarize(year = "2020",
```

```
            count = n(),
            avgComp = mean(AvgComp),
            minComp = min(AvgComp),
            maxComp = max(AvgComp))

comp_by_job <- bind_rows(comp_by_job_2017,
                         comp_by_job_2018,
                         comp_by_job_2019,
                         comp_by_job_2020)

# Plot the graph now
ggplot(comp_by_job, aes(x = JobCategory, y = avgComp/1000, fill = year)) +
  geom_bar(stat = "identity", position = "dodge") +
  ylab("Average Compensation (in 1000 USD)") +
  xlab("Job Category") +
  scale_fill_brewer(palette = "Pastel1") +
  geom_text(aes(label = round(avgComp/1000)), vjust = -0.2, size = 3,
            position = position_dodge(0.9))
```



The above graph shows the following:

1. The survey for 2018 when compensation ranges were introduced for the first time is not received well. There are unexpected values which are lower than usual. People have converted their salaries in USD from other currencies and put in the survey which has brought down the aggregated values. The country of origin being USA with data science salaries in 0-10,000 is unusual.
2. The ML Engineer related options were not given to the surveys of 2018 and 2019. However the salaries available for 2017 and 2020 shows the field is paying very well and desirable.

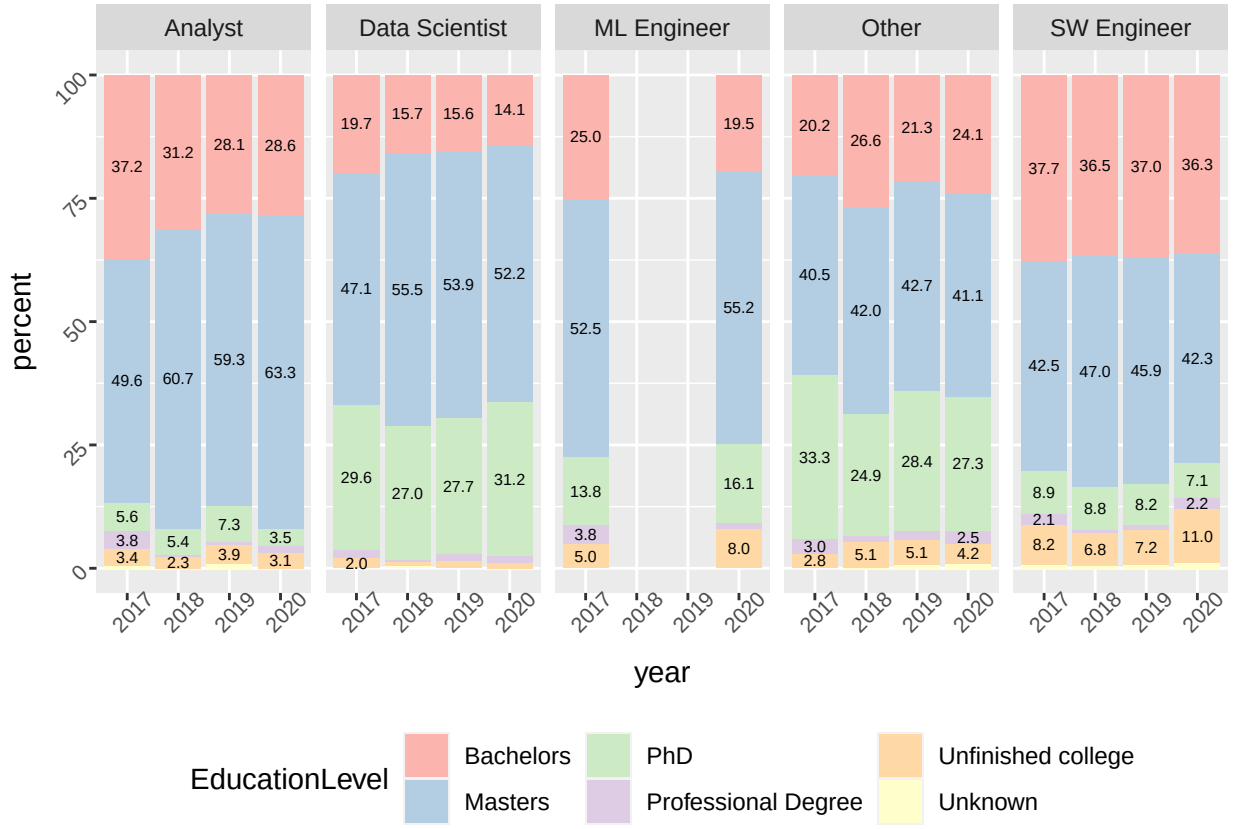
3. Close to the ML Engineer roles is the Data Scientist role, which has shown consistently high compensation over the years.
4. Other fields related to data science are not paid as high. The bucket of *Other* category has diverse salary ranges but not significant sample size which shows lesser jobs (perhaps needing niche skills) emerging in those categories.

4.3 Effect of formal education in the fields

Next we analyze the trends of level of education of people working in the field by different job categories.

```
generateEducationByJob <- function(df, jobCategory, yearVal) {
  ndf <- df %>%
    filter(JobCategory == jobCategory) %>%
    group_by(EducationLevel) %>%
    summarize(year = yearVal, count = n(), JobCategory = jobCategory) %>%
    ungroup() %>%
    arrange(desc(EducationLevel)) %>%
    mutate(percent = count * 100 / sum(count),
           ypos = cumsum(percent)- 0.5*percent)

  return(ndf)
}

ed_by_year <- bind_rows(
  generateEducationByJob(comp_2017, "Analyst", "2017"),
  generateEducationByJob(comp_2017, "Data Scientist", "2017"),
  generateEducationByJob(comp_2017, "ML Engineer", "2017"),
  generateEducationByJob(comp_2017, "SW Engineer", "2017"),
  generateEducationByJob(comp_2017, "Other", "2017"),
  generateEducationByJob(comp_2018, "Analyst", "2018"),
  generateEducationByJob(comp_2018, "Data Scientist", "2018"),
  generateEducationByJob(comp_2018, "ML Engineer", "2018"),
  generateEducationByJob(comp_2018, "SW Engineer", "2018"),
  generateEducationByJob(comp_2018, "Other", "2018"),
  generateEducationByJob(comp_2019, "Analyst", "2019"),
  generateEducationByJob(comp_2019, "Data Scientist", "2019"),
  generateEducationByJob(comp_2019, "ML Engineer", "2019"),
  generateEducationByJob(comp_2019, "SW Engineer", "2019"),
  generateEducationByJob(comp_2019, "Other", "2019"),
  generateEducationByJob(comp_2020, "Analyst", "2020"),
  generateEducationByJob(comp_2020, "Data Scientist", "2020"),
  generateEducationByJob(comp_2020, "ML Engineer", "2020"),
  generateEducationByJob(comp_2020, "SW Engineer", "2020"),
  generateEducationByJob(comp_2020, "Other", "2020")
)

ggplot(data=ed_by_year, aes(x=year, y=percent, fill=EducationLevel)) +
  geom_bar(stat="identity")+
  geom_text(aes(y = ypos,
                label = case_when(
                  percent > 2 ~ sprintf("%.1f", round(percent, 1)),
                  TRUE ~ ""
                )), color = "black", size = 2) +
  scale_fill_brewer(palette = "Pastel1") +
```

```
facet_wrap(JobCategory ~ ., nrow = 1, ncol = 5) +
theme(legend.position = "bottom",
      axis.text.x = element_text(size=8, angle=45),
      axis.text.y = element_text(size=8, angle=45))
```



From the results above we see that:

1. The industry is predominantly filled with people with Bachelors or higher education.
2. The maximum degree holders have Master's degree and for the roles of Data Scientist, ML Engineer or Other related fields (Research Scientist, Statistician, etc) there is a higher percentage of PhDs. This shows the utility for Masters or PhD to be good in the field.
3. The number of people with unfinished college degrees is highest for software engineering and lowest for Data Scientist positions.

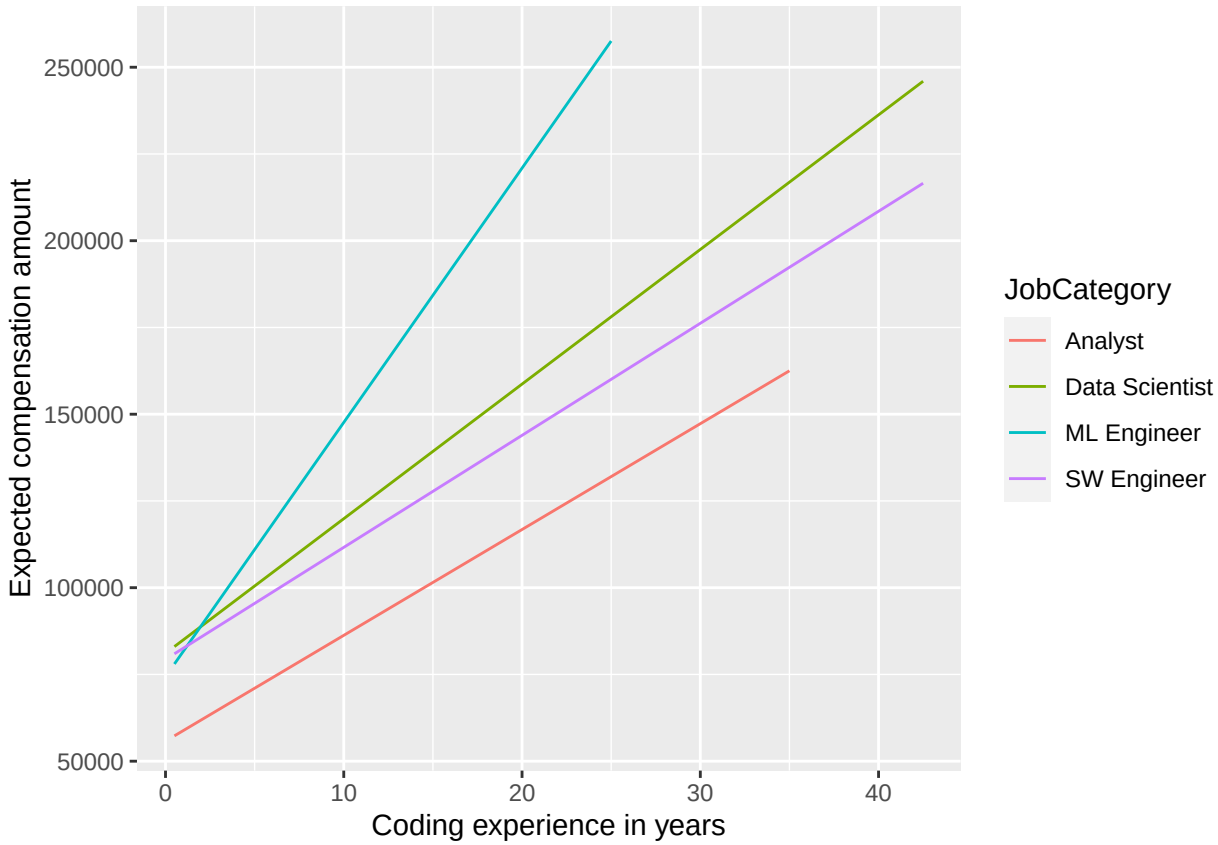
All these clearly establish the utility of higher education being high for pursuing long term career in data science and related fields.

4.4 Predict compensation vs year of coding experience

We will try to show how important coding experience is for different job categories by building a linear model of years of coding experience and compensation respectively. As both were collected as intervals we are going to map it to mid point on real line. This is an approximation to get a general trend and is understandably not a rigid predictor. Meaning the trend assessment will be accurate but ranges can vary by an error margin.

First, we will generate the compensations and coding experiences from previously generated *comp_20XX* data frames. We will only consider people who are working and earning for modeling purposes.

```r
codingDurationApproximation <- function(range) {
  value <- case_when(
    range %in% c("Less than a year", "< 1 year", "< 1 years") ~ 0.5,
    range %in% c("1 to 2 years", "1-2 years") ~ 1.5,
    range %in% c("3 to 5 years", "3-5 years") ~ 4,
    range %in% c("5-10 years","6 to 10 years") ~ 7.5,
    range %in% c("More than 10 years","10-20 years") ~ 15,
    range %in% c("20+ years","20-30 years") ~ 25,
    range == "30-40 years" ~ 35,
    range == "40+ years" ~ 42.5,
    TRUE ~ 0)

  return(value)
}

coding_comp_2017 <- comp_2017 %>%
  mutate(year = "2017", CodingApprox = codingDurationApproximation(CodingDuration),
         AvgComp = Compensation) %>%
  filter((CodingApprox > 0) & (!is.na(JobCategory))) %>%
  select(JobCategory, CodingApprox, AvgComp, year)
coding_comp_2018 <- comp_2018 %>%
  mutate(year = "2018", CodingApprox = codingDurationApproximation(CodingDuration)) %>%
  filter((CodingApprox > 0) & (!is.na(JobCategory))) %>%
  select(JobCategory, CodingApprox, AvgComp, year)
coding_comp_2019 <- comp_2019 %>%
  mutate(year = "2019", CodingApprox = codingDurationApproximation(CodingDuration)) %>%
  filter((CodingApprox > 0) & (!is.na(JobCategory))) %>%
  select(JobCategory, CodingApprox, AvgComp, year)
coding_comp_2020 <- comp_2020 %>%
  mutate(year = "2020", CodingApprox = codingDurationApproximation(CodingDuration)) %>%
  filter((CodingApprox > 0) & (!is.na(JobCategory))) %>%
  select(JobCategory, CodingApprox, AvgComp, year)

coding_comp <- bind_rows(coding_comp_2017,
                         coding_comp_2018,
                         coding_comp_2019,
                         coding_comp_2020)

plotCategory <- function(df, jobCategory) {
  dfjc <- df %>% filter(JobCategory == jobCategory)
  fit  <- lm(AvgComp ~ CodingApprox, data = dfjc)
  dfjc <- dfjc %>%
    mutate(pred = predict(fit))

  return(dfjc)
}

dfjcs <- bind_rows(plotCategory(coding_comp, "Analyst"),
                   plotCategory(coding_comp, "Data Scientist"),
                   plotCategory(coding_comp, "ML Engineer"),
                   plotCategory(coding_comp, "SW Engineer"))
ggplot(data = dfjcs, aes(x = CodingApprox, y = pred, color = JobCategory)) +
  geom_line() + xlab("Coding experience in years") + ylab("Expected compensation amount")
```



The grpah conclusively proves that having years of coding experience provides significant gains in career and compensation. The analysis highlights that:

1. With more years of experience the compensation is expected to grow. This does not include outlier companies but a trend on market level expectations.
2. All ML Engineers, Data Scientists and SW Engineers start around same salaries and lowest paid are analysts in the beginning.
3. Machine learning Engineers with more than 2-3 years experience tend to be paid higher and are highly sought after in industry.
4. Data scientists are high earners too and earn consistently more than Software Engineers and Analysts in long run.

4.5 What programming languages are used by different jobs?

To assess the programming languages and their closeness to various job profiles we will use correspondence analysis. It's a branch of multivariate analysis that allows appropriate visualization of qualitative variables spatially ie. with the help of a perception map.

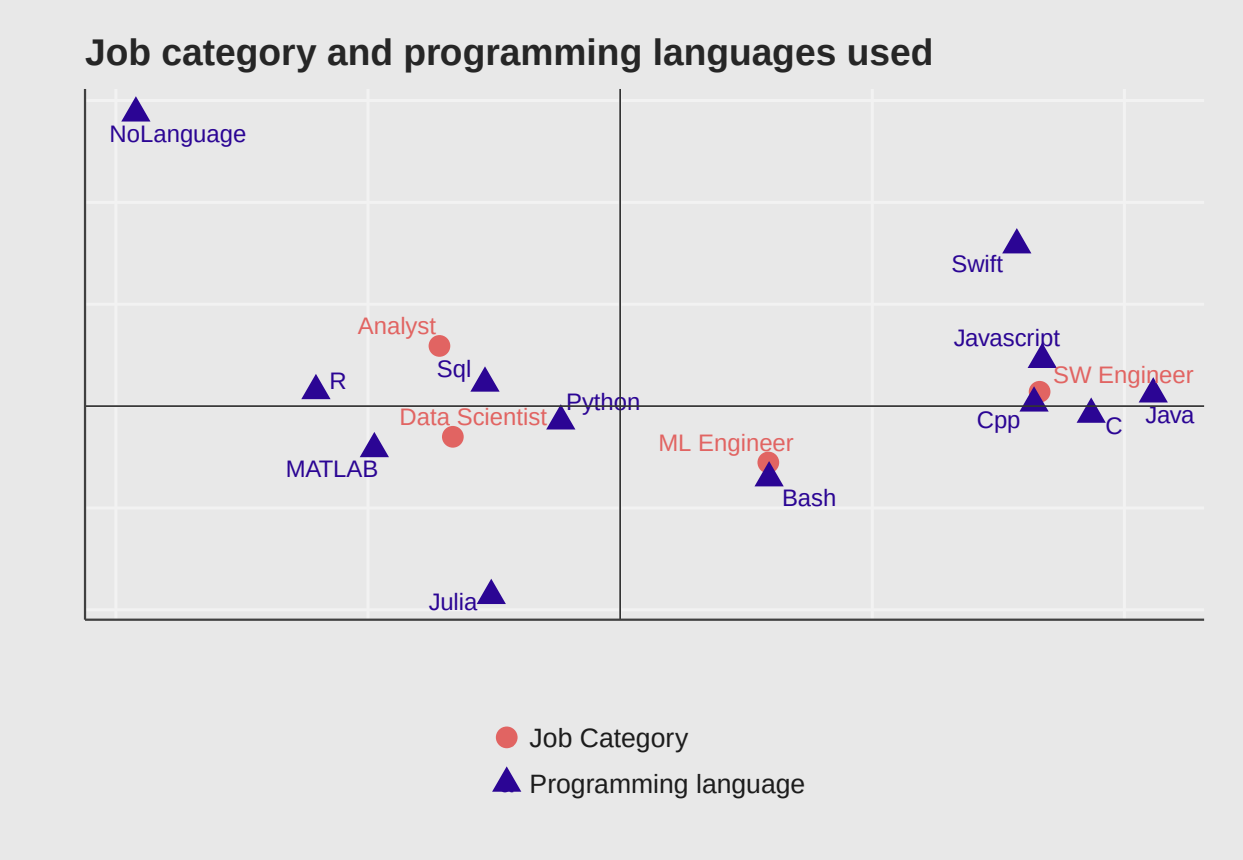
We will also focus on only 2020 survey for this part as the relevance of technology for preparation in next year is rapidly evolving year by year.

```
lang_df <- comp_2020 %>%
  group_by(JobCategory) %>%
  summarise(Python = length(which(!is.na(Python))),
            R = length(which(!is.na(R))),
            Sql = length(which(!is.na(Sql))),
            C = length(which(!is.na(C))),
```

```
            Cpp = length(which(!is.na(Cpp))),
            Java = length(which(!is.na(Java))),
            Javascript = length(which(!is.na(Javascript))),
            Julia = length(which(!is.na(Julia))),
            Swift = length(which(!is.na(Swift))),
            Bash = length(which(!is.na(Bash))),
            MATLAB = length(which(!is.na(MATLAB))),
            NoLanguage = length(which(!is.na(NoLanguage))), .groups = 'drop')

lang_df <- lang_df[-4,-1]
rownames(lang_df) <- c("Analyst", "Data Scientist", "ML Engineer", "SW Engineer")
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
ca_lang_df <- ca(lang_df)

ca_lang_df_2 <- as.data.frame(rbind(ca_lang_df$rowcoord, ca_lang_df$colcoord))
ca_lang_df_2$color <- ifelse(
  rownames(ca_lang_df_2) %in%
    c("Analyst", "Data Scientist", "ML Engineer", "SW Engineer"),
  "Job Category", "Programming language")

# Theme picked from stackoverflow.com and adapted for our problem
options(repr.plot.width=11, repr.plot.height=7)
ggplot(ca_lang_df_2, aes(Dim1, Dim2, fill = color, shape = color, color = color))+
  geom_point(size = 3.5)+
  geom_text_repel(aes(label = rownames(ca_lang_df_2)), size = 3)+
  geom_hline(yintercept = 0, size = 0.3, color = "gray20")+
  geom_vline(xintercept = 0, size = 0.3, color = "gray20")+
  scale_color_viridis_d(option = "plasma",  begin = 0.6, end = 0.05)+
  labs(title = "Job category and programming languages used", x = "", y = "")+
  theme_fivethirtyeight() +
  theme(legend.position = "bottom",
        legend.direction = "vertical",
        legend.key = element_rect(fill = "gray91"),
        legend.text = element_text(size = 10, colour = "gray11"),
        legend.title = element_text(size = 0),
        legend.background = element_rect(fill = "gray91"),
        axis.text = element_text(size = 0),
        axis.title = element_text(size = 10, colour = "gray15"),
        axis.line = element_line(size = 0.4, colour = "gray25"),
        plot.caption = element_text(color = "gray55", face = "bold", size = 9),
        plot.background = element_rect(fill = "gray91"),
        plot.title = element_text(size = 14, colour = "gray15"),
        plot.subtitle = element_text(size = 10, colour = "gray42"),
        strip.background = element_rect(fill = "gray75"),
        strip.text = element_text(size = 10, colour = "gray25", face = "bold"),
        panel.grid.major = element_line(colour = "gray95"),
        panel.background = element_rect(fill = "gray91"))
```



The figure brings out the following observations:

1. Software engineers and data science professionals don't really use the same programming languages. While former has more closeness to C/C++/Java/Swift/Javascript, the latter has more closeness to Python/R/SQL, etc. SQL though commonly underlies in Software engineering too, is more directly interacted by data science professionals.
2. ML Engineers use more bash programming and python than R. Data scientists seem to be close to Python, R and SQL more than ML Engineers.
3. Analysts seem to be more close to data scientists for programming languages and could reflect fade lines between the active job tools even though the analysts salaries are expected to be lower.

5 Possible biases

The analysis can have errors due to biases in data collection and filtering strategies used. Some of them possibly are:

1. The data is from one platform's survey. Even though it seems large, there are niche jobs such as Researchers and Statisticians who are not well represented. This hides the trends of some crucially related job profiles in data science.
2. Software engineers from large scale infrastructure companies could also not be well represented here. We expect the software engineers who participated in Kaggle's survey are those who work at an intersection of data science and engineering. Consequently, the compensation comparisons for niche engineers could be completely different.
3. The authenticity of data (salaries, years of experience, etc.) is not verifiable and would bring in errors. This would also bring in bias to our study.

6 Conclusion

We have analysed *Kaggle Machine Learning & Data Science Survey* from 2017-20 in this report. The analysis talks about the cleaning strategies used and the data parameters. To answer the questions we use data visualizations and bring out findings. We see the need for community to grow in terms of gender diversity. The compensation trends show the data science professionals to be doing well and the upside trends seems likely to continue. We also see that contrary to exceptions who claim success in data science field through no higher education, most people do hold a graduate level degree. Next, we have conclusively shown the importance of programming experience on each job profile and compared them across. Through correspondence analysis, we show the programming languages that are more relevant for data science professionals. Lastly, we talk about possible biases involved. However, even with the described biases, we do believe the market trends at national level would hold up. The values may move up or down due to errors, but the general trends seem aligned with some of the expectations in real world.