Stack Overflow Developer Survey Results 2019 Analyse

Florian Drinkler | Matrikelnummer: 6653948

Montag, 18. Januar 2021

Contents

1 Vorarbeiten **1**
1.1 Installieren der Pakete 1
1.2 Verbinden mit der Datenbank 2

2 Allgemeine Informationen **2**

3 Analyse **3**
3.1 Entwicklerprofil 3
3.2 Technologie 8
3.3 Beruf 11

4 Nacharbeiten **18**

5 Ergebnis **18**

Github: Github Repository Survey-Analysis

In dieser Arbeit wurden die Daten der Stack Overflow Developer Umfrage aus dem Jahr 2019 analysiert. Jedes Jahr werden Entwickler auf der ganzen Welt gefragt was Ihre Lieblingsprogrammiersprache ist bis hin zu Ihren Berufspräferenzen. 2019 haben fast 90.000 Entwickler teilgenommen.

1 Vorarbeiten

1.1 Installieren der Pakete

Um alle benötigten Pakete zu installieren kann der nachfolgende Code Snippet ausgeführt werden. Alle bereits installierten Pakete werden übersprungen.

```
list.of.packages <- c("ggplot2", "forcats", "scales", "dplyr", "reshape2", "DBI", "packcircles", "tidyr
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
```

Alle Pakete werden nun zur späteren Verwendung in das Markdown eingebunden.

```
# Import libraries
library(ggplot2)
library(forcats)
library(scales)
library(dplyr)
library(DBI)
library(tidyr)
library(packcircles) # https://github.com/mbedward/packcircles - Algorithms to find arrangements of non
library(reshape2) # https://github.com/hadley/reshape
```

```
library(ggrepel) # https://github.com/slowkow/ggrepel - ggrepel provides geoms for ggplot2 to repel ove
library(rworldmap) # https://github.com/AndySouth/rworldmap/ - R package for mapping global data
```

1.2 Verbinden mit der Datenbank

Da die Rohdaten der csv-Datei in einem geeigneten Format zur Verfügung gestellt sein müssen sowie Abfragen mit SQL möglich sein sollen, wurde eine .db-Datei mithilfe von sqlite3, Version 3.34.0, erstellt. Zu finden ist diese Datei in dem res Ordner.

Um die Daten mit Sql-Befehlen lesen und bearbeiten zu können, wird eine Verbindung mit der Datenbank erstellt.

```
# Connect to database
con <- dbConnect(RSQLite::SQLite(), "res/survey_results.db")
```

2 Allgemeine Informationen

In diesem Kapitel werden die grundlegenden Informationen der Umfragedaten untersucht.

```
dbListTables(con)
```

```
## [1] "survey"
```

Die Datenbank survey_results hat eine Tabelle namens “survey”.

```
survey <- dbReadTable(con, "survey")
```

```
dbListFields(con, "survey")
```

```
##  [1] "Respondent"             "MainBranch"             "Hobbyist"
##  [4] "OpenSourcer"            "OpenSource"             "Employment"
##  [7] "Country"                "Student"                "EdLevel"
## [10] "UndergradMajor"         "EduOther"               "OrgSize"
## [13] "DevType"                "YearsCode"              "Age1stCode"
## [16] "YearsCodePro"           "CareerSat"              "JobSat"
## [19] "MgrIdiot"               "MgrMoney"               "MgrWant"
## [22] "JobSeek"                "LastHireDate"           "LastInt"
## [25] "FizzBuzz"               "JobFactors"             "ResumeUpdate"
## [28] "CurrencySymbol"         "CurrencyDesc"           "CompTotal"
## [31] "CompFreq"               "ConvertedComp"          "WorkWeekHrs"
## [34] "WorkPlan"               "WorkChallenge"          "WorkRemote"
## [37] "WorkLoc"                "ImpSyn"                 "CodeRev"
## [40] "CodeRevHrs"             "UnitTests"              "PurchaseHow"
## [43] "PurchaseWhat"           "LanguageWorkedWith"     "LanguageDesireNextYear"
## [46] "DatabaseWorkedWith"     "DatabaseDesireNextYear" "PlatformWorkedWith"
## [49] "PlatformDesireNextYear" "WebFrameWorkedWith"     "WebFrameDesireNextYear"
## [52] "MiscTechWorkedWith"     "MiscTechDesireNextYear" "DevEnviron"
## [55] "OpSys"                  "Containers"             "BlockchainOrg"
## [58] "BlockchainIs"           "BetterLife"             "ITperson"
## [61] "OffOn"                  "SocialMedia"            "Extraversion"
## [64] "ScreenName"             "SOVisit1st"             "SOVisitFreq"
## [67] "SOVisitTo"              "SOFindAnswer"           "SOTimeSaved"
## [70] "SOHowMuchTime"          "SOAccount"              "SOPartFreq"
## [73] "SOJobs"                 "EntTeams"               "SOComm"
## [76] "WelcomeChange"          "SONewContent"           "Age"
## [79] "Gender"                 "Trans"                  "Sexuality"
```

```
## [82] "Ethnicity"              "Dependents"             "SurveyLength"
## [85] "SurveyEase"
```

Mithilfe der Funktion dbReadTable, welche die ganze Tabelle ausgibt, und DbListFields, welche die Spaltennamen der Tabelle ausgibt, kann man sich einen Überblick über die Tabelle verschaffen. Für Anschauungszwecke wird die Datenbank lokal gespeichert. Dieser wird dann entfernt, bis sie nicht mehr gebraucht wird.

```
dim(survey)
```

```
## [1] 88883    85
```

Die Tabelle hat 85 Dimensionen und 88.883 Merkmale.

```
#str(survey)
```

Mit der Funktion "str()" kann man sich eine Überblick über die Datentypen und Dateninhalten der Spalten machen. Aufgrund des langen Outputs wird die Funktion auskommentiert.

```
rm(survey)
```

Da die Tabelle nicht mehr lokal gebraucht wird, wir sie aus dem Speicher entfernt.

3 Analyse

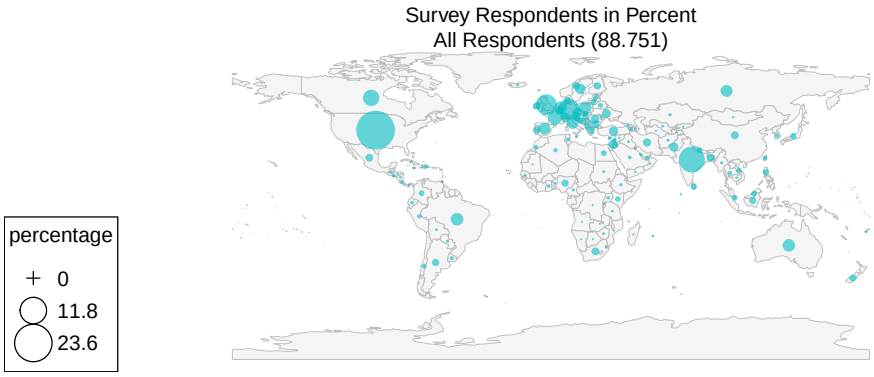
3.1 Entwicklerprofil

```
# Return countries with number of participants
respondents <- dbGetQuery(con,
                          "SELECT Country, COUNT(*) AS 'Number of participants'
                          FROM survey
                          WHERE Country IS NOT NULL
                          GROUP BY Country
                          ORDER BY 'Number of participants' DESC;")

# Convert to dataframe
respondents_df <- as.data.frame(respondents)

# Add percentage
responses <- sum(respondents_df[["Number of participants"]])
respondents_df[["percentage"]] <- (respondents_df[["Number of participants"]] / responses * 100) %>% ro

# Convert countries to an internal map, used for plotting
sPDF <- joinCountryData2Map(respondents_df,
                            joinCode = "NAME",
                            nameJoinColumn = "Country")
```

```
## 173 codes from your data successfully matched countries in the map
## 6 codes from your data failed to match with a country code in the map
## 72 codes from the map weren't represented in your data
```

```
# Plot map
mapBubbles(dF = sPDF,
           nameZSize = "percentage",
           nameZColour = adjustcolor("#00bfc4", alpha.f = 0.6),
           legendPos = "bottomleft",
           landCol = "#F5F5F5",
           addLegend = TRUE )
```

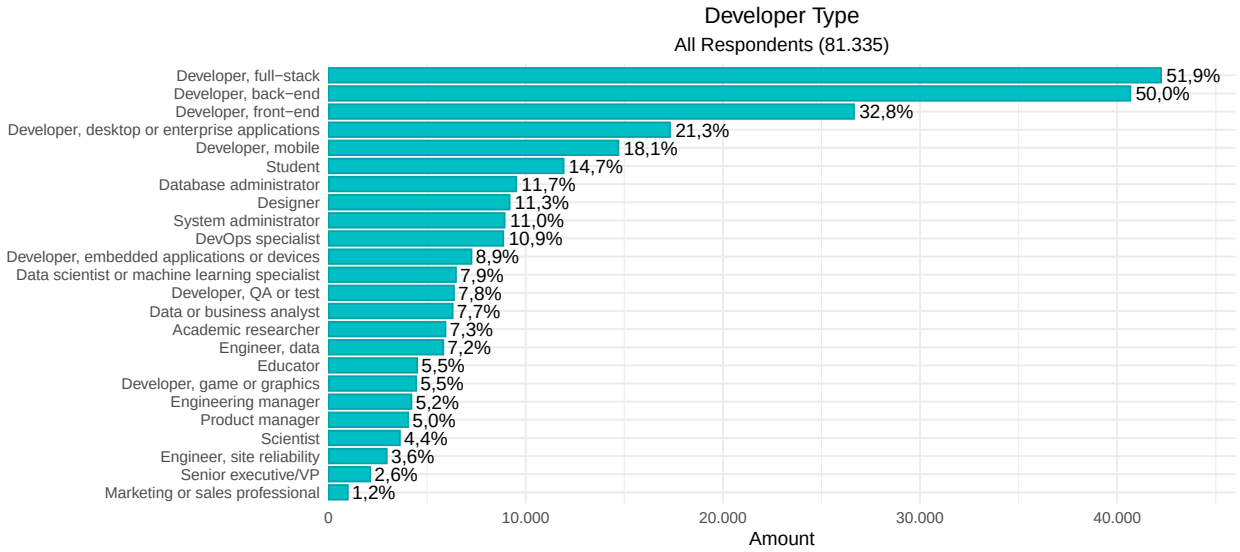
```
mtext("Survey Respondents in Percent",side=3,line=0.5)
mtext(paste("All Respondents (", responses %>% format(decimal.mark = ",", big.mark = "."), ")", sep = "
```



Diese Karte zeigt die Herkunft aller Teilnehmer. Der jeweilige Kreis eines Landes zeigt den prozentualen Anteil der Gesamtteilnehmer an. Dies heißt, dass der Kreis die Anzahl der Teilnehmer jedes Landes angibt. Je größer der Kreis, umso größer ist die Anzahl der Teilnehmer.
Am meisten Teilnehmer stammen aus der USA. Dort sind es ungefähr 23%. Aus Deutschland kommen ungefähr 6%. Aus Indien stammen 10%, jedoch hat dieses Land sehr viele Einwohner.

```
# Return devtypes without nulls
devtypes <- dbGetQuery(con,
                       "SELECT DevType
                       FROM survey
                       WHERE DevType IS NOT NULL;")

# Preparate data
devtypes_split <- strsplit(devtypes$DevType, split = ";")

# Flatten list and convert to dataframe
devtypes_df <- as.data.frame(unlist(devtypes_split))
colnames(devtypes_df) <- colnames(devtypes)

# Sort by frequency of values in descending order
devtypes_df <- mutate(devtypes_df, DevType = fct_rev(fct_infreq(DevType)))

# Plot data
ggplot(devtypes_df, aes(x = DevType)) +
  geom_bar(color = "#00a4a8", fill = "#00bfc4", width = 0.8) +
  coord_flip() +
  theme_minimal() +
  labs(title = "Developer Type",
       subtitle = lengths(devtypes) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", ., ")", sep = ""),
```

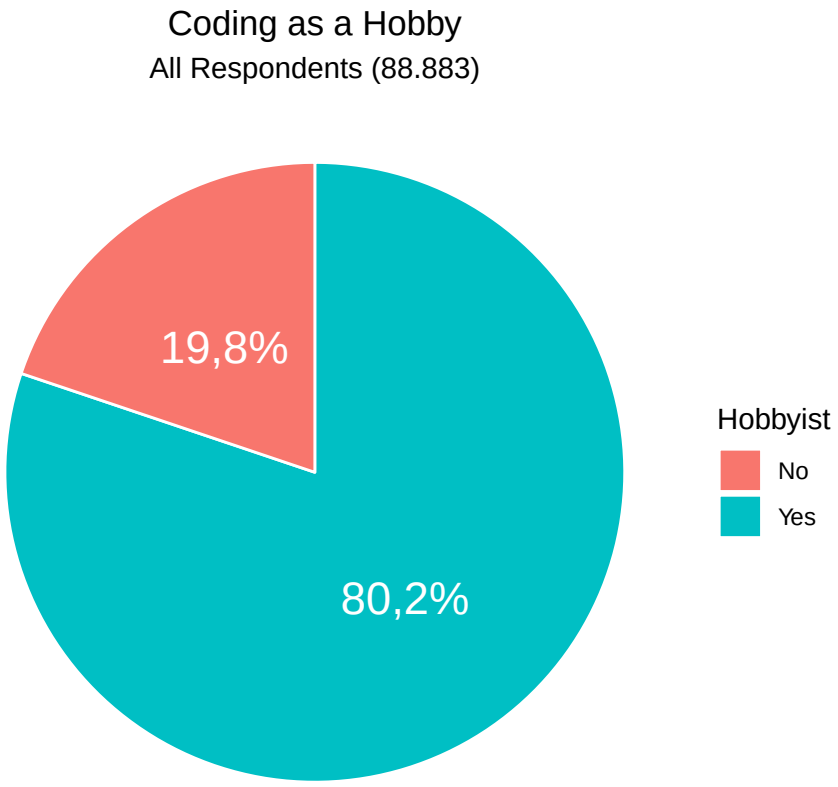
```
      y = "Amount") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.y = element_blank()) +
  scale_y_continuous(limits = c(0, 46000), breaks = seq(0, 46000, 10000), expand = c(0,0), labels = scal
  geom_text(aes(label = stat((count/lengths(devtypes)) %>% scales::percent(decimal.mark = ",", accuracy
```



In diesem Balkendiagramm werden alle verschiedenen Entwicklertypen, nach der Häufigkeit absteigend, angezeigt. Hinter jedem Balken steht eine Prozentzahl, die die Häufigkeit des Typen angibt. Ein einzelner Entwickler konnte mehrere Typen wählen.
Über 50% der Teilnehmer erkennen sich als Full-Stack sowie Backend Entwickler an. Im Durchschnitt hat jeder Entwickler drei Typen ausgewählt. Ebenso gibt es starke Korrelationen zwischen Berufen wie DevOps specialist und Site reliability engineer oder Database administrator und System administrator.
Diese Daten wurden von allen Ländern genommen, daher kann es zu Unterscheidungen in den einzelnen Ländern kommen.

```
# Return hobbyist grouped by yes or no as dataframe
hobbyist <- dbGetQuery(con,
                       "SELECT Hobbyist, COUNT(*) AS Amount
                       FROM survey
                       WHERE Hobbyist IS NOT NULL
                       GROUP BY Hobbyist
                       ORDER BY Amount DESC;") %>%
  as.data.frame()

# add Percentage and position of labels
hobbyist <- hobbyist %>%
  mutate(prop = hobbyist$Amount / sum(hobbyist$Amount) * 100) %>%
  mutate(ypos = cumsum(prop) - 0.5*prop)

# Plot data
ggplot(hobbyist, aes(x = "", y = prop, fill = Hobbyist)) +
  geom_bar(stat="identity", width = 1, color = "white") +
  coord_polar("y", start = 0) +
```

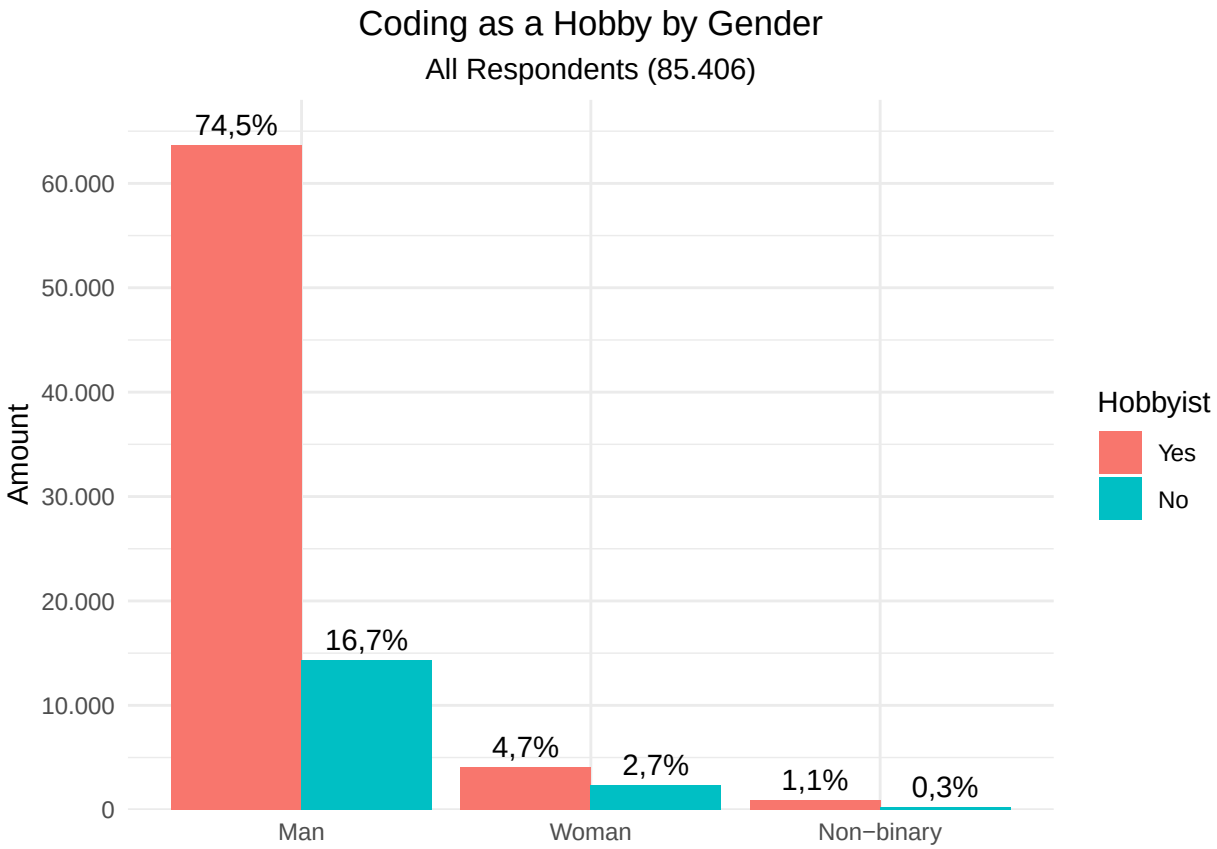
```
  theme_void() +
  labs(title = "Coding as a Hobby",
       subtitle = sum(hobbyist$Amount) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", ., ")", sep = "")) +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5)) +
  geom_text(aes(y = ypos, label = prop %>% format(digits = 3, decimal.mark = ",") %>% paste("%", sep =
```



In diesem Kreisdiagramm wird angezeigt ob ein Entwickler das Programmieren als Hobby ansieht. Dies bedeutet, dass der Programmierer in seiner Freizeit, nach der Arbeit, an eigenen Projekten arbeitet.
Mehr als 80% aller befragten Entwickler sagen das Programmieren ein Hobby von Ihnen ist. Fast 20% entwickeln nur in der Arbeit.

```
# Return hobbyist grouped by yes or no as dataframe
hobby_gender <- dbGetQuery(con,
                           "SELECT Gender, Hobbyist, COUNT(*) AS Amount
                           FROM survey
                           WHERE Hobbyist IS NOT NULL
                           AND Gender IS NOT NULL
                           GROUP BY Hobbyist, Gender
                           ORDER BY Amount DESC;") %>%
  as.data.frame()

# Combine diverse genders amount
```

```r
others <- hobby_gender %>%
  slice(5:14) %>%
  group_by(Hobbyist) %>%
  summarise(Gender = "Non-binary", Amount = sum(Amount), .groups = 'drop') %>%
  arrange(desc(Amount)) %>%
  relocate(Gender, .before = Hobbyist)

# Remove others from dataframe
hobby_gender <- hobby_gender[-c(5:14), ]

# Combine both dataframes
hobby_gender <- rbind(hobby_gender, others)

# Plot data
ggplot(hobby_gender, aes(factor(reorder(Gender, -Amount)), Amount, fill = fct_rev(Hobbyist))) +
  geom_bar(stat = "identity", position = "dodge") +
  theme_minimal() +
  scale_fill_manual("Hobbyist", values = c("Yes" = "#f8766d", "No" = "#00bfc4")) +
  labs(title = "Coding as a Hobby by Gender",
       subtitle = sum(hobby_gender$Amount) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", ., ")", sep = ""),
       fill = "Hobbyist") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x = element_blank()) +
  scale_y_continuous(limits = c(0, 68000), breaks = seq(0, 68000, 10000), expand = c(0, 0), labels = sca
  geom_text(aes(label = (Amount/sum(Amount)) %>% scales::percent(decimal.mark = ",", accuracy = 0.1)), p
```

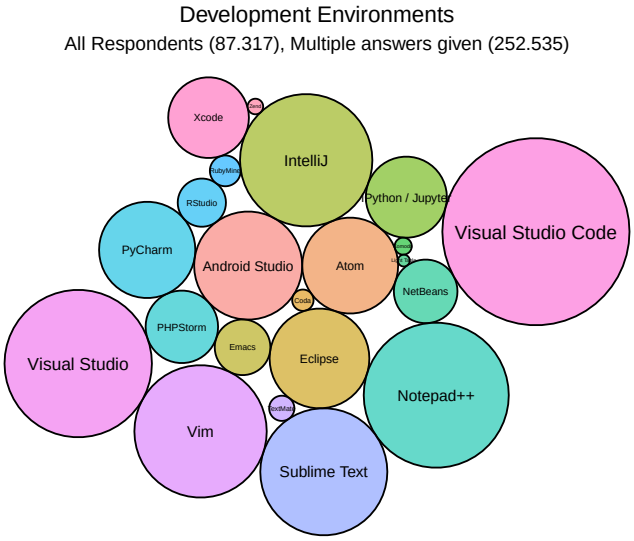


In diesem Säulendiagramm werden wird die Frage ob ein Teilnehmer Programmieren als Hobby hat, auf die jeweiligen Geschlechter aufgeteilt. Man kann bereits an der Länge der Säulen erkennen, dass viel weniger Frauen an der Umfrage teilgenommen haben. Dies liegt daran, da es bereits viel weniger Frauen in der IT-Brache bzw. Software Entwicklung gibt. Die Prozentzahl auf den Säulen ist von der Teilnehmerzahl abhängig.
Männer ähneln dem bereits vorher gezeigten Kreisdiagramm. Bei Frauen ist das Programmieren als Hobby weniger verbreitet. Non-binaries sind am schwächsten Vertreten.

3.2 Technologie

```
# Return Development Environments
devEnviron <- dbGetQuery(con,
                         "SELECT DevEnviron
                         FROM survey
                         WHERE DevEnviron IS NOT NULL;") %>%
  as.data.frame()

# Get amount of respondents
respondents <- length(devEnviron$DevEnviron) %>%
  format(big.mark = ".", decimal.mark = ",")

# Subtract data from query
singleDevEnviron <- strsplit(devEnviron$DevEnviron, split = ";") %>%
  unlist() %>%
  as.data.frame()
```

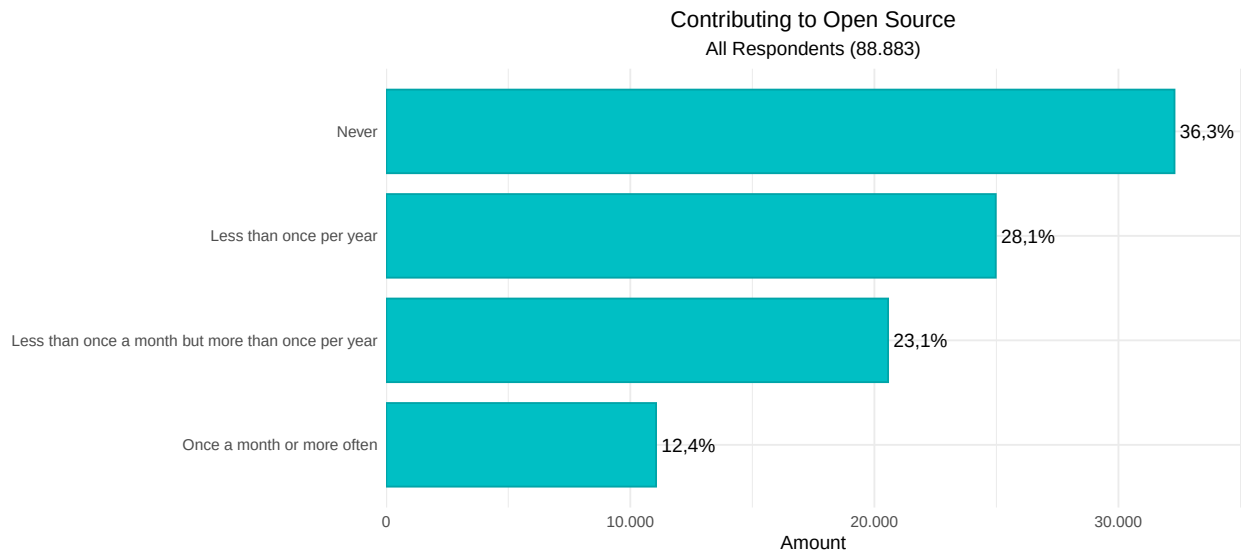
```r
# Rename column
names(singleDevEnviron)[1] <- "devEnviron"

# Group dev Environments and get amount
devEnviron <- singleDevEnviron %>% group_by(devEnviron) %>% tally()

# Generate the layout. This function return a dataframe with one line per bubble. It gives its center (
packing <- circleProgressiveLayout(devEnviron$n, sizetype = "area")

# Add the packing information to the initial data frame
devEnviron <- cbind(devEnviron, packing)

# The next step is to go from one center + a radius to the coordinates of a circle that is drawn by a m
devEnviron.gg <- circleLayoutVertices(packing, npoints = 50)

# Plot data
ggplot() +
  geom_polygon(data = devEnviron.gg, aes(x, y, group = id, fill=as.factor(id)), colour = "black", alpha
  geom_text(data = devEnviron, aes(x, y, size = n, label = devEnviron)) +
  scale_size_continuous(rang = c(1,4)) +
  labs(title = "Development Environments",
       subtitle = sum(devEnviron$n) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", respondents, "), Multiple answers given (", ., ")", sep = "")) +
  theme_void() +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        legend.position = "none") +
  coord_equal()
```



Aufgrund vom persönlichen Interesse wurde die Häufigkeit der verwendeten Entwicklungsumgebungen als Bubblediagramm dargestellt. In jedem Kreis steht die zugehörige Entwicklungsumgebung. Je größer der Kreis, umso mehr war diese unter den Teilnehmern vertreten. Jeder Teilnehmer durfte mehrere Entwicklungsumgebungen auswählen.
Man sollte jedoch beachten das z.B. Visual Studio Code mehrere Plattformen und Programmierensprachen

unterstützt. Android Studio im Gegensatz wird nur zur Programmierung von Androidapps verwendet, genauso wie Xcode nur zur Apps- und Anwendungsprogrammierung für Apple Geräte verwendet wird. Überrascht hat mich die Anzahl von Notepad++, Sublime Text und Vim. Notepad++ und Sublime Text, weil es sich hier jeweils nur um einen Texteditor handelt. Ich vermute das diese, aufgrund der Mehrfachauswahl, nicht als Hauptumgebung verwendet wird, sondern für kleinere Aufgaben. Vim ist sehr komplex und schwierig, was eine kleinere Anzahl zu vermuten lassen würde.

```r
# Return countries with number of participants
opensourcer <- dbGetQuery(con,
                          "SELECT OpenSourcer, COUNT(*) AS Amount
                          FROM survey
                          WHERE OpenSourcer IS NOT NULL
                          GROUP BY OpenSourcer
                          ORDER BY Amount DESC;")

# Convert to dataframe
opensourcer_df <- as.data.frame(opensourcer)

# Plot data
ggplot(opensourcer_df, aes(x = reorder(OpenSourcer, Amount), y = Amount)) +
  geom_bar(stat = "identity", color = "#00a4a8", fill = "#00bfc4", width = 0.8) +
  coord_flip() +
  theme_minimal() +
  labs(title = "Contributing to Open Source",
       subtitle = sum(opensourcer_df$Amount) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", ., ")", sep = "")) +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.y = element_blank()) +
  scale_y_continuous(limits = c(0, 35000), breaks = seq(0, 35000, 10000), expand = c(0, 0), labels = sca
  geom_text(aes(label = (Amount/sum(Amount)) %>% scales::percent(decimal.mark = ",", accuracy = 0.1)), h
```

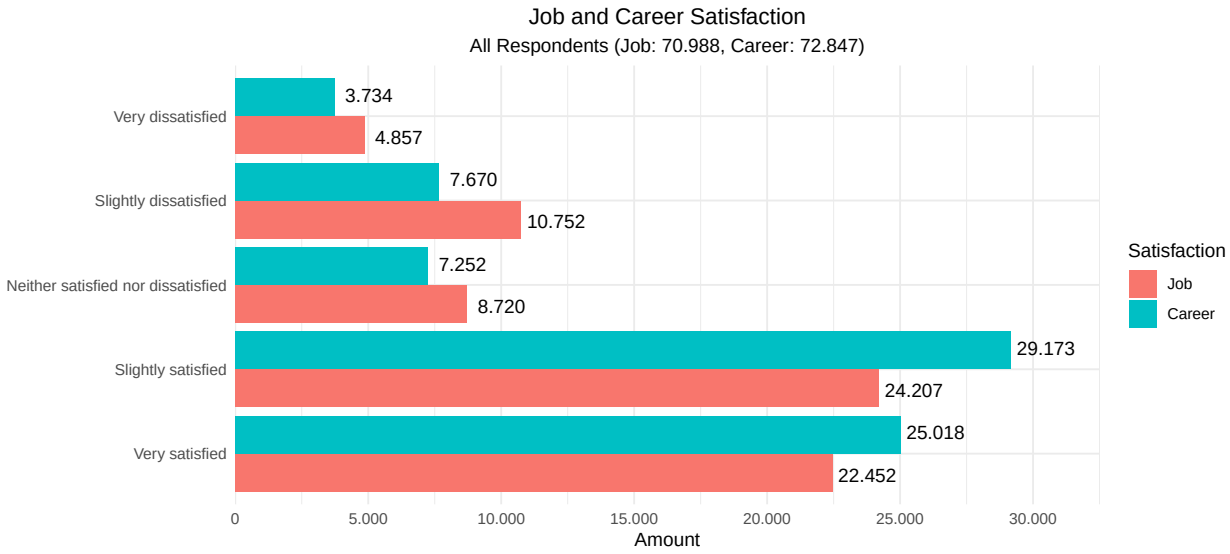


Alle Teilnehmer wurden befragt ob diese an Open-Source Projekten teilnehmen. Aus diesem Balkendiagramm wird deutlich, dass die Mehrheit weniger an Open-Source Projekten einen Beitrag leisten.

Von oben nach unten wird die Bereitschaft für Open-Source Projekte angezeigt. Nur 12,4% aller Befragten leisten mindestens einmal im Monat einen Beitrag zu Open-Source Projekten.
65% der Teilnehmer leisten mindestens einmal im Jahr einen Beitrag. Der Beitrag zu Open-Source Projekten variiert nach der Programmiersprache eines Projektes.

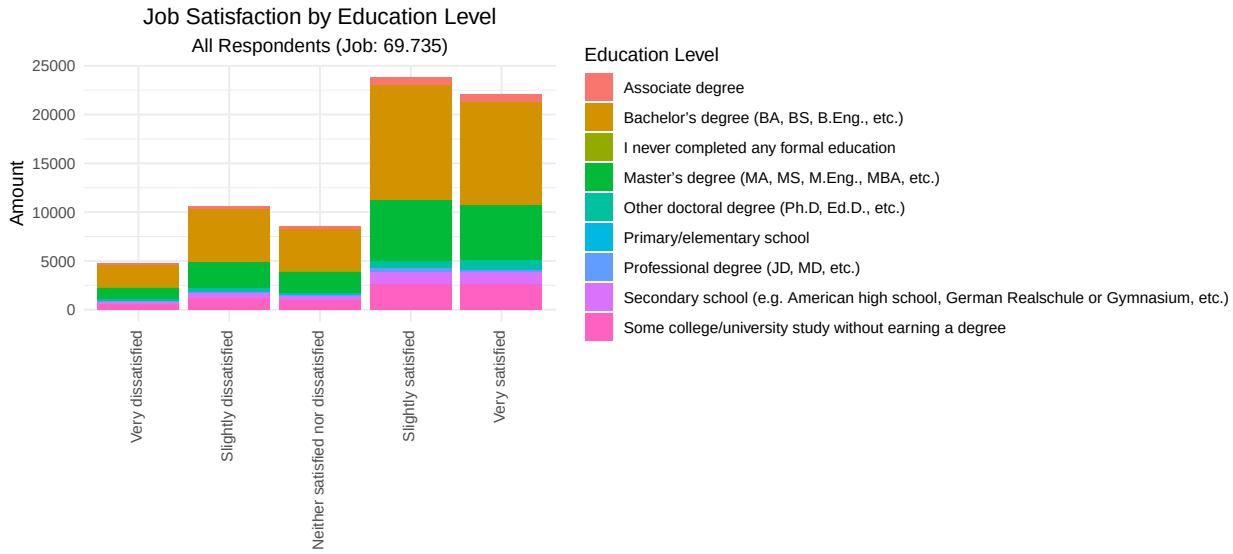
3.3 Beruf

```
# Return JobSat grouped as dataframe
jobsat <- dbGetQuery(con,
                     "SELECT JobSat, COUNT(*) AS Job
                     FROM survey
                     WHERE JobSat IS NOT NULL
                     GROUP BY JobSat
                     ORDER BY Job DESC;") %>%
  as.data.frame()

# Return CareerSat grouped as dataframe
carsat <- dbGetQuery(con,
                     "SELECT CareerSat, COUNT(*) AS Career
                     FROM survey
                     WHERE CareerSat IS NOT NULL
                     GROUP BY CareerSat
                     ORDER BY Career DESC;") %>%
  as.data.frame()

# Reorder x-axis
jobsat$JobSat <- fct_rev(factor(jobsat$JobSat, levels = c("Very dissatisfied", "Slightly dissatisfied",
carsat$CareerSat <- fct_rev(factor(carsat$CareerSat, levels = c("Very dissatisfied", "Slightly dissatis

# Combine both dataframes
satisfaction <- cbind(jobsat, carsat %>% select(Career))

# Get sums for subtitle
jobsum <- sum(satisfaction$Job) %>% format(big.mark = ".", decimal.mark = ",")
careersum <- sum(satisfaction$Career) %>% format(big.mark = ".", decimal.mark = ",")

# Plot data
ggplot(melt(satisfaction, id.vars = "JobSat"), aes(x = JobSat, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "dodge")+
  coord_flip() +
  theme_minimal() +
  scale_fill_manual("Satisfaction", values = c("Job" = "#f8766d", "Career" = "#00bfc4")) +
  labs(title = "Job and Career Satisfaction",
       subtitle = paste("All Respondents (Job: ", jobsum, ", Career: ", careersum, ")", sep = ""),
       y = "Amount") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.y = element_blank()) +
  scale_y_continuous(limits = c(0, 32500), breaks = seq(0, 32500, 5000), expand = c(0, 0), labels = scal
  geom_text(aes(label = value %>% format(big.mark=".", decimal.mark=",")), hjust = -0.1, position = posi
```

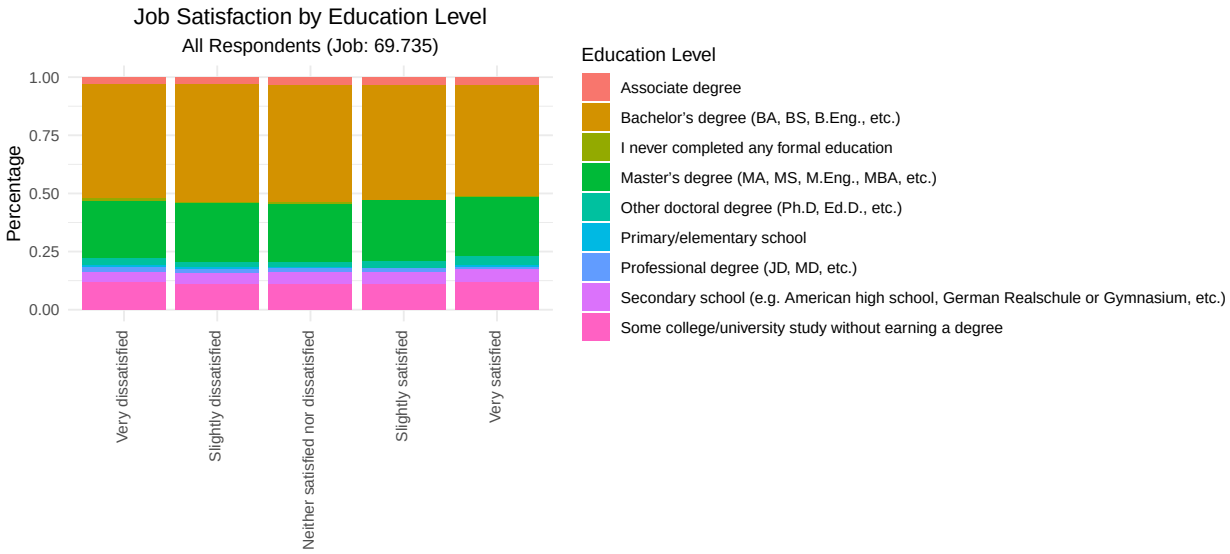


In diesem Balkendiagramm wird die Berufszufriedenheit mit der Karrierezufriedenheit verglichen. Aufsteigend wird die Zufriedenheit angezeigt, von sehr unzufrieden zu sehr zufrieden. 72.847 Teilnehmer haben eine Angabe über Ihre Karrierezufriedenheit angegeben wohingegen nur 70.998 eine Angabe zur Berufszufriedenheit angegeben haben.
Zu sehen ist, dass die Teilnehmer im Allgemeinen unzufriedener mit Ihrem jetztigen Beruf sind als mit Ihrer Karriere. Es kann vermutet werden, dass Programmierer unzufriedener mit Ihrer Arbeit sind oder nicht Ihr volles Potenzial ausgenutzt werden kann, wohingegen bei der Karriere noch Aussicht auf Verbesserung ist und somit Hoffnung besteht.

```r
# Return JobSat by degree grouped as dataframe
jobsat_by_degree <- dbGetQuery(con,
                      "SELECT JobSat, EdLevel, COUNT(*) AS Amount
                      FROM survey
                      WHERE JobSat IS NOT NULL
                      AND EdLevel IS NOT NULL
                      GROUP BY JobSat, EdLevel;") %>%
  as.data.frame()

# Reorder x-axis
jobsat_by_degree$JobSat <- factor(jobsat_by_degree$JobSat, levels = c("Very dissatisfied", "Slightly dis

# Plot data
ggplot(jobsat_by_degree, aes(fill = EdLevel, x = JobSat, y = Amount)) +
  geom_bar(position = "stack", stat = "identity") +
  theme_minimal() +
  scale_fill_discrete(name = "Education Level") +
  labs(title = "Job Satisfaction by Education Level",
       subtitle = paste("All Respondents (Job: ", sum(jobsat_by_degree$Amount) %>% format(big.mark = ".
       y = "Amount") +
  theme(axis.text.x = element_text(angle=90, vjust=0.3, hjust=1),
        plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x = element_blank())
```



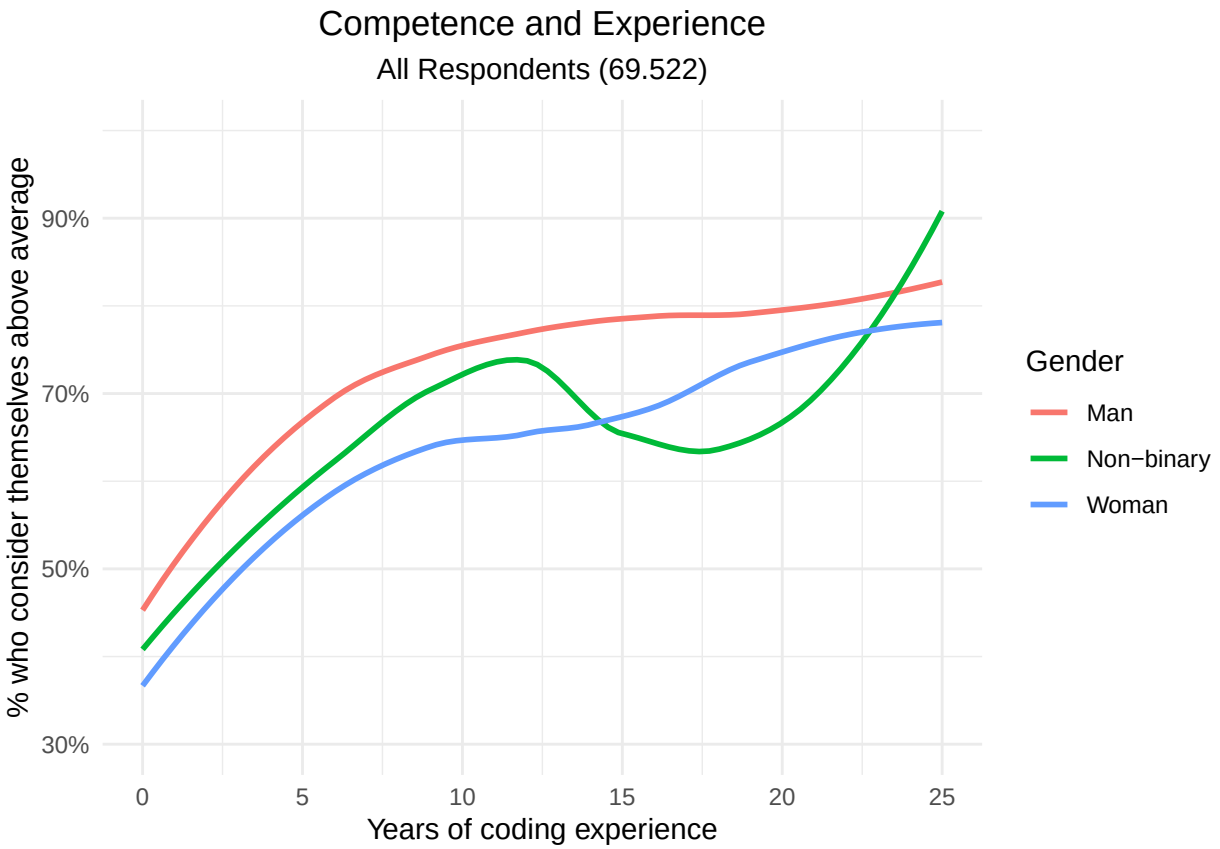

```r
# Return JobSat by degree grouped as dataframe
jobsat_by_degree <- dbGetQuery(con,
                     "SELECT JobSat, EdLevel, COUNT(*) AS Amount
                     FROM survey
                     WHERE JobSat IS NOT NULL
                     AND EdLevel IS NOT NULL
                     GROUP BY JobSat, EdLevel;") %>%
  as.data.frame()

# Reorder x-axis
jobsat_by_degree$JobSat <- factor(jobsat_by_degree$JobSat, levels = c("Very dissatisfied", "Slightly dis

# Plot data
ggplot(jobsat_by_degree, aes(fill = EdLevel, x = JobSat, y = Amount)) +
  geom_bar(position = "fill", stat = "identity") +
  theme_minimal() +
  scale_fill_discrete(name = "Education Level") +
  labs(title = "Job Satisfaction by Education Level",
       subtitle = paste("All Respondents (Job: ", sum(jobsat_by_degree$Amount) %>% format(big.mark = ".
       y = "Percentage") +
  theme(axis.text.x = element_text(angle=90, vjust=0.3, hjust=1),
        plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x = element_blank())
```



In den oberen beiden gruppierten Säulendiagrammen wird die Berufszufriedenheit in Abhängigkeit mit dem Ausbildungsgrad verglichen.
Beide gehen von "Sehr unzufrieden" bis hin zu "Sehr zufrieden".
Im ersten Diagramm wird die Verteilung der Zufriedenheits nochmals deutlicher. Jedoch kann man noch nicht aussagen welche Gruppe zufriedener ist.
Daher wird das zweite Diagramm rangezogen. Hierbei wird prozentual die Anzahl der Ausbildungsgraden mit der jeweiligen Gesamtanzahl jeder Zufriedenheit verglichen. Es wird deutlich, dass die Zufriedenheit nicht von dem Ausbildungsgrad abhängig ist.
Beispielsweise, Orange/Braun sind Teilnehmer mit einem Bachelor. Im ersten Diagramm könnte man davon ausgehen, dass diese zufriedener sind. Jedoch aus dem zweiten Diagramm erkennt man, dass diese über alle Zufriedenheiten gleich verteilt sind.

```r
# Get Information about Gender, Competence and Experience
yig <- dbGetQuery(con,
                  "SELECT YearsCodePro, ImpSyn, Gender
                  FROM survey
                  WHERE YearsCodePro IS NOT NULL
                  AND ImpSyn IS NOT NULL
                  AND Gender IS NOT NULL
                  AND (ImpSyn = 'A little above average'
                  OR ImpSyn = 'Far above average');") %>%
  as.data.frame()

# Get respondants of each gender
ga <- dbGetQuery(con,
                 "SELECT YearsCodePro, ImpSyn, Gender
                 FROM survey
                 WHERE YearsCodePro IS NOT NULL
                 AND ImpSyn IS NOT NULL
                 AND Gender IS NOT NULL;") %>%
  as.data.frame()

# Transform YearsCodePro into numbers
yig$YearsCodePro[yig$YearsCodePro == "Less than 1 year"] <- 0
yig$YearsCodePro[yig$YearsCodePro == "More than 50 years"] <- 50
```

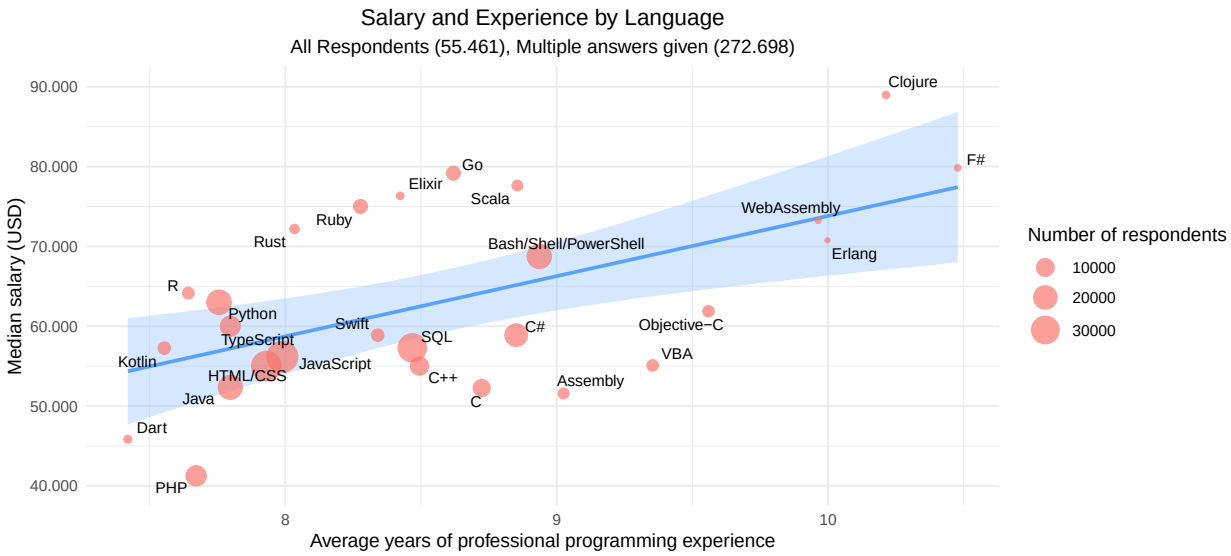
```r
yig <- transform(yig, YearsCodePro = as.numeric(YearsCodePro))

ga$YearsCodePro[ga$YearsCodePro == "Less than 1 year"] <- 0
ga$YearsCodePro[ga$YearsCodePro == "More than 50 years"] <- 50
ga <- transform(ga, YearsCodePro = as.numeric(YearsCodePro))

# Group genders
yig$Gender[yig$Gender != "Man" & yig$Gender != "Woman"] <- "Non-binary"
ga$Gender[ga$Gender != "Man" & ga$Gender != "Woman"] <- "Non-binary"

# Get Amounts
yig <- yig %>% group_by(YearsCodePro, Gender) %>% tally()
ga <- ga %>% group_by(YearsCodePro, Gender) %>% tally()

# Merge Amounts into new data frame
colnames(ga)[3] <- "N"
df <- merge(yig, ga, by.ga = "N", all=TRUE)
df <- df[complete.cases(df[ ,]),]

# Plot data
ggplot(df, aes(x = YearsCodePro, y = n/N, color = Gender)) +
  geom_smooth(aes(color = Gender), se=F, method = "loess", formula = "y ~ x") +
  labs(title = "Competence and Experience",
       subtitle = sum(ga$N) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", ., ")", sep = ""),
       x = "Years of coding experience",
       y = "% who consider themselves above average") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5)) +
  scale_y_continuous(limits = c(0.30, 1), breaks = seq(0.30, 1, 0.2), labels = scales::percent_format(bi
  scale_x_continuous(limits = c(0, 25), breaks = seq(0, 25, 5))
```



Die Befragten wurden ausdrücklich gebeten, sich selbst für Ihre jahrelange Erfahrung zu bewerten. Man sieht jedoch Meinungsverschiedenheiten. Neue Entwickler bewerten sich aufgrund Ihrer Erfahrung viel schlechter als Überdurchschnittlich. Nach ungefähr 10 Jahren lässt dies nach. Bei jungen Entwicklern kann man das Betrugssyndrom vermuten, da die Kompetenz schnell anstieg und danach nicht mehr stark wächst. Außerdem wurden Männer viel schneller selbstbewusster als geschlechtsspezifische Minderheiten.
Die extreme Kurve bei Non-binaries kann man auf zu wenig Daten bzw. falschen Daten zurückführen.
Nach 25 Jahren wurde der Graph beendet, da die nachfolgenden Daten zu ungenau sind.

```
# Get Information about Salary, Working experience and used Languages
saebl <- dbGetQuery(con,
                    "SELECT ConvertedComp, YearsCodePro, LanguageWorkedWith
                    FROM survey
                    WHERE ConvertedComp IS NOT NULL
                    AND YearsCodePro IS NOT NULL
                    AND LanguageWorkedWith IS NOT NULL;") %>%
  as.data.frame()

# Separate Languages
saebl_separated <- saebl %>% separate_rows(LanguageWorkedWith, sep = ";")

# Transform Coding experience column into numbers
saebl_separated$YearsCodePro[saebl_separated$YearsCodePro == "Less than 1 year"] <- 1
saebl_separated$YearsCodePro[saebl_separated$YearsCodePro == "More than 50 years"] <- 50
saebl_separated <- transform(saebl_separated, YearsCodePro = as.numeric(YearsCodePro))
```

```r
# Get amount of respondents for each language
LanguagesCount <- saebl_separated %>% count(LanguageWorkedWith)

# Group by Languages and get median of Salary and average working experience
saebl_separated <- saebl_separated %>% group_by(LanguageWorkedWith) %>% summarise(medianConvertedComp =

# Join both dataframes
saebl_all<-cbind(saebl_separated, LanguagesCount[!names(LanguagesCount) %in% names(saebl_separated)])

# Remove "Other(s)" language from data
saebl_all <- subset(saebl_all, LanguageWorkedWith!="Other(s):")

# Plot data
ggplot(saebl_all, aes(x = meanYearsCodePro, y = medianConvertedComp, size = n)) +
  geom_smooth(method=lm , color="#5AA2F9", fill="#99c7ff", se=TRUE, show.legend = FALSE, formula = "y ~
  geom_point(alpha = 0.7, color = "#f8766d") +
  scale_size(range = c(1, 8), name="Number of respondents") +
  theme_minimal() +
  labs(title = "Salary and Experience by Language",
       subtitle = sum(saebl_all$n) %>%
         format(big.mark = ".", decimal.mark = ",") %>%
         paste("All Respondents (", length(saebl$LanguageWorkedWith) %>% format(big.mark = ".", decimal
       x = "Average years of professional programming experience",
       y = "Median salary (USD)") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5)) +
  scale_y_continuous(
    limits = c(40000, 90000),
    breaks = seq(40000, 90000, 10000),
    labels = scales::number_format(big.mark = ".", decimal.mark = ",")) +
  geom_text_repel(label = saebl_all$LanguageWorkedWith, size = 3)
```



In diesem Diagramm wird das Gehalt und der Erfahrung mit der jeweiligen Programmiersprache verglichen. Die blaue Linie in der Mitte ist der Durchschnitt. Programmiersprachen die über dieser Linie sind, z.B. Rust, Go oder auch Scala werden auch ohne weniger Jahre an Erfahrung mehr bezahlt als Programmiersprachen

wie PHP, Assembler. Diese werden auch mit vielen Jahren an Erfahrung weniger bezahlt.
Die Größe der Kreise repräsentiert die Anzahl an Entwicklern die diese Programmiersprache verwenden im Gegensatz zu anderen Programmiersprachen.

4 Nacharbeiten

Schließen der Datenbankverbindung.

```
# Disconnect database
dbDisconnect(con)
```

5 Ergebnis

Diese Daten basieren auf der Stack Overflow Umfrage aus dem Jahre 2019, die vom 23 Januar bis 14 Februar 2020 stattgefunden hat. An der Umfrage haben 88.883 Entwickler aus 179 Ländern teilgenommen.

Die erstellen Diagramme und Graphen liefern einen Einblick in die Daten der Umfrage. Es können Rückschlüsse, wie welche Programmiersprache an Beliebtheit erhält, wo wie viel Geld verdient wird, welche Berufe am beliebtesten sind und die Erfahrungen von Teilnehmern mit Programmierung, gezogen werden.

In dieser Programmieraufgabe ist mir die Programmiersprache R ans Herz gewachsen. Nach einem langsamen Einstieg gelang es früh die ersten Balkendiagramme zu erstellen. Nach und nach sind mehrere Praktiken bekannt geworden und man konnte somit schneller und besser Programmieren.
Bei den Diagrammen wurde auf eine korrekte Schreibweise und auf Kleinigkeiten der Diagramme geachtet. Beispielsweise wurden an den x und y Achsen Punkte bei großen Zahlen verwendet oder Kommas anstatt Punkte verwendet, sowie auf die Formatierung von den Daten geachtet.
Ich freue mich schon auf die Stack Overflow Umfrage 2020 wo ich auf jeden Fall einen Blick hineinwerfen werde.