

# OccupationalProfiles\_Analysis

*Barbara Hofer*

*31 7 2019*

## Contents

License . . . . .	2
Acknowledgements . . . . .	2
Metadata . . . . .	2
Data . . . . .	3
Frequency of duties over the profiles . . . . .	6
Cross section through the profiles per duty . . . . .	7

## License

not yet specified

## Acknowledgements

This work was supported by the Sector Skill Alliance project EO4GEO - <http://www.eo4geo.eu> Many thanks to all EO4GEO partners who contributed occupational profiles.

## Metadata

Required libraries and runtime environment description.

```
library("here")
library("dplyr")
library("stringr")
library("DT")
library("data.table")
library("formattable")
library("htmltools")
library("devtools")
library("knitr")
```

## Data

The data used in the analysis are the .csv file named `Profiles_inputdata_180719.csv`. This file has been generated based on occupational profiles that have been contributed by partners of the EO4GEO project. The file contains the duties and tasks with in some cases newly assigned labels of profiles related to Remote sensing and GIS workforce.

```
# to set the path to the directory where the code resides
here::here("Profile_extendedanalysis_test", "Github_Material_July2019")
```

```
## [1] "C:/Owncloud/EO4GEO_project_2018/Profile_extendedanalysis_test/Github_Material_July2019/Profile_
```

```
list.files(path = ".")
```

```
## [1] "Duties_skills"
## [2] "Duties_Task_Renaming"
## [3] "Duties_trends"
## [4] "OccupationalProfile_Collection_Instructions"
## [5] "OccupationalProfiles_Analysis_July2019.html"
## [6] "OccupationalProfiles_Analysis_July2019.pdf"
## [7] "OccupationalProfiles_Analysis_July2019.Rmd"
## [8] "Profiles_inputdata_300719.csv"
## [9] "readme.md"
```

```
profiles <- read.csv2("Profiles_inputdata_300719.csv", header = TRUE, sep = ";")
```

```
#---
```

```
# coherence of strings in the duties and tasks this step is only done
# for the first manipulation of duties and tasks; manual adaptations
# have been done afterwards if statement checks, whether the colum of
# duty.man is empty and executes the code then
```

```
#---
```

```
if (sum(profiles$duty.man == "") == length(profiles$duty)) {
```

```
# for duties: to lower case, ignore everything in brackets, & replace
# by and, delete white spaces at the end of string
```

```
profiles$duty.man <- str_to_lower(profiles$duty, locale = "en")
profiles$duty.man <- str_replace(profiles$duty.man, "\\(.*\\)", "")
profiles$duty.man <- str_replace(profiles$duty.man, "&", "and")
profiles$duty.man <- str_trim(profiles$duty.man)
```

```
# for tasks: to lower case, ignore everything in brackets, & replace by
# and, delete white spaces at the end of string
```

```
profiles$task.man <- str_to_lower(profiles$original.task.name, locale = "en")
profiles$task.man <- str_replace(profiles$task.man, "\\(.*\\)", "")
profiles$task.man <- str_replace(profiles$task.man, "&", "and")
profiles$task.man <- str_trim(profiles$task.man)
```

```
# write.csv2(profiles, 'Profiles_inputdata_manipulated15042019.csv')
```

```
}
```

```
# number of original duties and manipulated duties:
nduty <- length(unique(profiles$duty))
nduty.man <- length(unique(profiles$duty.man))
```

```
# number of (original) duties
nduty
```

```
## [1] 51
```

```
# number of manipulated duties
nduty.man
```

```
## [1] 19
```

```
# table summarizing the manipulation of the duties
dutymanes <- profiles %>% group_by(profiles$duty, profiles$duty.man) %>%
  summarize()

datatable(dutymanes)
```

Show  entries Search:

	profiles\$duty	profiles\$duty.man
1	Accuracy assessment	quality control and validation
2	administration	administration
3	application development	IT infrastructure and application development
4	communication	communication and interaction with stakeholders
5	data acquisition	data acquisition
6	Data acquisition	data acquisition
7	data analysis	data analysis
8	Data analysis	data analysis
9	Data Analysis	data analysis
10	Data analysis	data analysis

Showing 1 to 10 of 51 entries Previous  2 3 4 5 6 Next

```
ntasks <- length(unique(profiles$original.task.name))
ntasks.man <- length(unique(profiles$task.man))
```

```
# number of (original) tasks
ntasks
```

```
## [1] 406
```

```
# number of manipulated tasks
ntasks.man
```

```
## [1] 295
```

```
# generate a table for the original and manipulated tasks
tasknames <- profiles %>% group_by(profiles$original.task.name, profiles$task.man) %>%
  summarize()
datatable(tasknames)
```

Show  entries

Search:

	profiles\$original.task.name	profiles\$task.man
1	accounting	accounting
2	accuracy metrics - goodness of fit process	accuracy metrics - goodness of fit process
3	acquire feedback from users	acquire user/customer feedback
4	acquire feedback from users (user validation)	acquire user/customer feedback
5	acquire feedback from users and incorporate feedback	acquire user/customer feedback
6	acquire feedback from users and incorporate feedback	incorporate user/customer feedback
7	Airborne data pre-processing (radiometric correction, geometric correction, coregistration)	airborne data pre-processing
8	analyse and interpret processes	analyse and interpret processes
9	analyse data	analyse data
10	analyse needs of users (incl. Order info)	analyse user needs

Showing 1 to 10 of 430 entries

Previous  2 3 4 5 ... 43 Next

```
# I do a grouping by profile and duties and summarize the values the
# grouping is necessary because I have a duty entry for every task
# (row); so I summarize by original.profile to have a non-redundant
# list of duties
profduty <- profiles %>% group_by(profiles$duty.man, profiles$profile.name) %>%
  summarize()

# here I count the number of different duties in the created variable
nduty <- count((profduty[, 1]))

# here the frequency of duties is calculated (in percent of total)
dutyfreq <- (nduty[, 2]/nprofiles) * 100

# add frequency to duties
nduty["dfreq"] <- dutyfreq[, 1]

# inner join of profiles with nduty data frame (column names need to be
# identical)
```

```

nduty <- nduty %>% rename(duty.man = "profiles$duty.man")
profiles <- merge(profiles, nduty, by = "duty.man")

# order the profiles according to the frequency of duties

oprofiles <- profiles %>% group_by(profiles$duty.man, profiles$dfreq) %>%
  summarize()

# renaming of columns
oprofiles <- oprofiles %>% rename(duty.man = "profiles$duty.man")
oprofiles <- oprofiles %>% rename(dfreq = "profiles$dfreq")

# round percentage values
oprofiles$dfreq <- round(oprofiles$dfreq, 0)

# ordering by dfreq
oprofiles <- oprofiles[order(oprofiles$dfreq, decreasing = T), ]

```

## Frequency of duties over the profiles

The frequency of duties is provided in the following table:

```
datatable(oprofiles)
```

Show  entries

Search:

	duty.man	dfreq
1	professional development	100
2	data analysis	91
3	project management	91
4	deliverables/product preparation	82
5	data management	73
6	communication and interaction with stakeholders	64
7	data acquisition	64
8	quality control and validation	64
9	projects conception	55
10	research and development	55

Showing 1 to 10 of 19 entries

Previous  2 Next

## Cross section through the profiles per duty

In the following, two directories are created and filled with HTML table that include the tasks of single duties. The tasks are thereby either colored according to the indicate skill level that is required or according to trends that were identified. This results in two collections of 19 tables each that provide the main output of this analysis.

```
# I build on the 19 duties and extract tasks for each profile in the
# original order I need the set of duties to iterate over in each
# iteration, I need to identify the profiles containing the duties and
# select the tasks of the duty then display the tasks according to the
# given order
dir.create("Duties_skills")
dir.create("Duties_trends")

nduties <- length(unique(profiles$duty.man))

dutylist <- unique(profiles$duty.man)

dutylist <- str_replace(dutylist, "/", "or")
profiles$duty.man <- str_replace(profiles$duty.man, "/", "or")

menge <- data.frame()

remove(proftasks)
remove(proftasksp)
remove(proftasksskill)
remove(proftaskstrend)

for (d in 1:length(dutylist)) {
  proftasks <- data.frame()
  proftaskstrend <- data.frame()
  proftasksskill <- data.frame()

  menge <- profiles[profiles$duty.man == dutylist[d], ]
  menge <- menge[order(menge$profile.source, menge$profile.name, menge$task.order),
    ]

  sourceprof <- menge %>% group_by(menge$profile.source, menge$profile.name) %>%
    summarize()
  sourceprof <- sourceprof %>% rename(profile.source = "menge$profile.source")
  sourceprof <- sourceprof %>% rename(profile.name = "menge$profile.name")

  for (p in 1:nrow(sourceprof)) {

    taskset <- select(menge[menge$profile.name == sourceprof$profile.name[p],
      ], "task.man")
    tasksettrend <- select(menge[menge$profile.name == sourceprof$profile.name[p],
      ], "future.trend")
    tasksetskill <- select(menge[menge$profile.name == sourceprof$profile.name[p],
      ], "skill.level..s.t.c.")

    taskdf <- data.frame(taskset)
```

```

taskdf <- transpose(taskdf)

taskdft <- data.frame(tasksettrend)
taskdft <- transpose(taskdft)

taskdfs <- data.frame(tasksetskill)
taskdfs <- transpose(taskdfs)

profilename <- merge(paste(sourceprof$profile.source[p]), paste(sourceprof$profile.name[p]))
proftasksp <- merge(profilename, taskdf)
proftasks <- rbindlist(list(proftasks, proftasksp), fill = TRUE)

proftaskstrend <- rbindlist(list(proftaskstrend, taskdft), fill = TRUE)

proftasksskill <- rbindlist(list(proftasksskill, taskdfs), fill = TRUE)

}

# write.csv2(proftasks, paste('Duties_skills/', dutylist[d], '.csv',
# sep=''), row.names = TRUE)

# Visualizing skills of tasks resulting in HTML tables
ntasks <- ncol(proftasks)
nskills <- ncol(proftasksskill)

colnames(proftasksskill) <- paste("SV", 1:nskills, sep = "")

taskvis <- cbind(proftasks, proftasksskill)

taskvis[is.na(taskvis)] <- "--"

n = (ncol(taskvis) - 2)/2
# this seems to work for the SV part of the list!
SVcolumns <- do.call(list, lapply(1:n, function(i) {
  return(FALSE)
}))
names(SVcolumns) <- paste("SV", 1:n, sep = "")

Vcolumns <- do.call(list, lapply(1:n, function(i) {
  return(formatter("span", style = ~style(`background-color` = ifelse(taskvis[[paste("SV",
i, sep = "")]] == "s", "lemonchiffon", ifelse(taskvis[[paste("SV",
i, sep = "")]] == "c", "lightcoral", ifelse(taskvis[[paste("SV",
i, sep = "")]] == "t", "mistyrose", "white"))))))))

}))
names(Vcolumns) <- paste("V", 1:n, sep = "")

SVandVformat <- c(SVcolumns, Vcolumns)

```



```

# formattable(taskvis, align=rep('l', n), SVandVformat)

# format_table provide the html version of the table (otherwise the
# function is called formattable)
html_header = "
<head>
<meta charset=\"utf-8\">
<meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">
<link rel=\"stylesheet\" href=\"https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css\">
</head>
<body>
"

html_table = format_table(taskvis, align = rep("l", n), SVandVformat)

write(paste(html_header, html_table, sep = ""), paste("Duties_skills/",
  dutylist[d], ".html", sep = ""))

# Hier brauche ich noch ein paar oder um die Begriffe abzugrenzen und
# noch weitere Farben und Kontrolle wichtig um zu sehen, ob die Trends
# richtig eingefärbt werden Visualizing trends of tasks resulting in
# HTML tables
ntasks <- ncol(proftasks)
ntrends <- ncol(proftaskstrend)

colnames(proftaskstrend) <- paste("SV", 1:ntrends, sep = "")

taskvistrend <- cbind(proftasks, proftaskstrend)

taskvistrend[is.na(taskvistrend)] <- "-"

n = (ncol(taskvistrend) - 2)/2
# this seems to work for the SV part of the list!
SVcolumns <- do.call(list, lapply(1:n, function(i) {
  return(FALSE)
}))
names(SVcolumns) <- paste("SV", 1:n, sep = "")

Vcolumns <- do.call(list, lapply(1:n, function(i) {
  return(formatter("span", style = ~style(`background-color` = ifelse(taskvistrend[[paste("SV",
    i, sep = "")]] == "ias", "cadetblue", ifelse(taskvistrend[[paste("SV",
    i, sep = "")]] == "pas", "lightsteelblue", ifelse(taskvistrend[[paste("SV",
    i, sep = "")]] == "analysis ready data", "lightyellow", ifelse(taskvistrend[[paste("SV",
    i, sep = "")]] == "automation of image analysis", "azure",
    "white"))))))))

}))
names(Vcolumns) <- paste("V", 1:n, sep = "")

SVandVformat <- c(SVcolumns, Vcolumns)

formattable(taskvis, align = rep("l", n), SVandVformat)

```

```

    # format_table provide the html version of the table (otherwise the
    # function is called formattable)
    html_header = "
<!DOCTYPE html>
<html>
<head>
<meta charset=\"utf-8\">
<meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">
<link rel=\"stylesheet\" href=\"https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css\">
</head>
<body>
"
    html_footer = "</html>"

    html_table = format_table(taskvistrend, align = rep("l", n), SVandVformat)

    write(paste(html_header, html_table, html_footer, sep = ""), paste("Duties_trends/",
        dutylist[d], ".html", sep = ""))

    remove(proftasks)
    remove(proftasksp)
}

```