Expense reports

Diana

12/13/2021

Overview

This is an Expense reports. Data analysis below allows to view spending on various categories, predict expenses. Also it will be calculated total budget.

Prepare phase

Import data sets and required libraries for analysis.

First it needs to import data sets include information about spending for several month (Jul - till Dec). Using library 'readr' allows import csv files. Using library 'tidyverse' to apply most of useful R functions.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(readr)
library(dplyr)
library(modelr)
Jul <- read_csv("Jul.csv")
```

```
## New names:
## * `` -> ...13
## * `` -> ...14
## * `` -> ...15
## * `` -> ...16
```

```
## Rows: 31 Columns: 16
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): Food, ...15
## dbl (13): Home, Other, Closes, Tehcnics, Diime, Transport, Urgent, Bills, Ap...
## lgl  (1): ...13
```

```
##
## 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.
```

```
Aug <- read_csv("Aug.csv")
```

```
## New names:
## * `` -> ...13
## * `` -> ...14
```

```
## Rows: 35 Columns: 14
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Food
## dbl (12): Home, Other, Closes, Tehcnics, Diime, Transport, Urgent, Bills, Ap...
## lgl  (1): ...13
```

```
##
## 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.
```

```
Sep_Oct <- read_csv("Sep_Oct.csv")
```

```
## Rows: 45 Columns: 12
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Food
## dbl (11): Home, Other, Closes, Tehcnics, Diime, Transport, Urgent, Bills, Ap...
```

```
##
## 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.
```

```
Nov <- read_csv("Nov.csv")
```

```
## Rows: 39 Columns: 12
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Food
## dbl (11): Home, Other, Closes, Tehcnics, Diime, Transport, Urgent, Bills, Ap...
```

```
##
## 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.
```

```
Dec <- read_csv("Dec.csv")
```

```
## Rows: 51 Columns: 12
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Food
## dbl (11): Home, Other, Closes, Tehcnics, Diime, Transport, Urgent, Bills, Ap...
```

```
##
## 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.
```

View, cleaning and transform data

Let look at the data, what kind of values present, data types, check same or not the data format. After reading csv file the result shows that some data sets don't have the same number of columns: - 'Jul' has 16,

'Aug' has 14, the rest have 12

Using function 'colnames' to compare the different sets of columns

```
colnames(Jul)
```

```
##  [1] "Food"       "Home"       "Other"      "Closes"     "Tehcnics"
##  [6] "Diime"      "Transport"  "Urgent"     "Bills"      "Apartments"
## [11] "Pills"      "Body"       "...13"      "...14"      "...15"
## [16] "...16"
```

```
colnames(Aug)
```

```
##  [1] "Food"       "Home"       "Other"      "Closes"     "Tehcnics"
##  [6] "Diime"      "Transport"  "Urgent"     "Bills"      "Apartments"
## [11] "Pills"      "Body"       "...13"      "...14"
```

```
colnames(Sep_Oct)
```

```
##  [1] "Food"       "Home"       "Other"      "Closes"     "Tehcnics"
##  [6] "Diime"      "Transport"  "Urgent"     "Bills"      "Apartments"
## [11] "Pills"      "Body"
```

Due to result it needs only the first 12 columns. The columns need to match perfectly before we can use a command to join them into one file for our data analysis later.

```
Jul <- Jul %>% select(c(Food,Home,Other,Closes,Tehcnics,Diime,Transport,Urgent,Bills,Apartments,Pills,B
Aug <- Aug %>% select(c(Food,Home,Other,Closes,Tehcnics,Diime,Transport,Urgent,Bills,Apartments,Pills,B
```

It needs also to have all data type as numeric. Later it will use statistical analysis. Due to import result: - the column 'Food' is string type. Let use the function 'as.double' to convert 'Food' to double type.

```
Jul$Food <- as.double(as.character(Jul$Food))
```

```
## Warning: NAs introduced by coercion
```

```
Aug$Food <- as.double(as.character(Aug$Food))
```

```
## Warning: NAs introduced by coercion
```

```
Sep_Oct$Food <- as.double(as.character(Sep_Oct$Food))
```

```
## Warning: NAs introduced by coercion
```

```
Nov$Food <- as.double(as.character(Nov$Food))
```

```
## Warning: NAs introduced by coercion
```

```
Dec$Food <- as.double(as.character(Dec$Food))
```

```
## Warning: NAs introduced by coercion
```

Exploratory Data Analysis

Summary result of all data

In this part we will analyse the combined data sets - the spending for all month. Using function 'bind_rows' to wrangle data in one data frame. Using function 'slice' to delete unnecessary rows (total rows from sheet). Using function 'summary' to overview data (values and statistics).

```
all_spending <- bind_rows(Jul, Aug, Sep_Oct, Nov, Dec)
all_spending <- all_spending %>%
  slice(-c(30, 31, 65,66,110,111,148,149,150)) %>%
```

```
  slice_head(n = 155)
summary(all_spending)
```

```
##       Food             Home             Other            Closes
##  Min.   : 1.39   Min.   :  0.000   Min.   : 1.000   Min.   :  3.00
##  1st Qu.: 5.00   1st Qu.:  3.425   1st Qu.: 1.670   1st Qu.: 10.25
##  Median : 8.45   Median :  7.800   Median : 3.000   Median : 18.00
##  Mean   :11.15   Mean   : 14.239   Mean   : 3.233   Mean   : 23.97
##  3rd Qu.:15.00   3rd Qu.: 13.500   3rd Qu.: 4.000   3rd Qu.: 30.15
##  Max.   :60.00   Max.   :140.000   Max.   :15.000   Max.   :100.00
##  NA's   :15      NA's   :111       NA's   :126      NA's   :129
##     Tehcnics          Diime           Transport         Urgent
##  Min.   : 20.00   Min.   : 1.000   Min.   : 1.20   Min.   : 20.00
##  1st Qu.: 53.75   1st Qu.: 2.000   1st Qu.: 3.00   1st Qu.: 38.30
##  Median : 87.50   Median : 3.500   Median : 7.00   Median : 70.00
##  Mean   : 86.75   Mean   : 6.736   Mean   :10.81   Mean   : 63.66
##  3rd Qu.:120.50   3rd Qu.:10.000   3rd Qu.:15.75   3rd Qu.: 80.00
##  Max.   :152.00   Max.   :25.000   Max.   :30.00   Max.   :110.00
##  NA's   :151      NA's   :141      NA's   :139     NA's   :150
##      Bills          Apartments        Pills            Body
##  Min.   : 1.00   Min.   :540.0   Min.   : 2.600   Min.   : 2.00
##  1st Qu.: 5.25   1st Qu.:583.1   1st Qu.: 5.000   1st Qu.: 5.00
##  Median : 6.00   Median :598.8   Median : 6.000   Median : 8.00
##  Mean   :10.05   Mean   :595.1   Mean   : 6.668   Mean   :10.76
##  3rd Qu.:15.00   3rd Qu.:610.8   3rd Qu.: 8.500   3rd Qu.:12.34
##  Max.   :30.00   Max.   :643.0   Max.   :12.500   Max.   :31.00
##  NA's   :145     NA's   :151     NA's   :146      NA's   :142
```

As we can see there are a lof NA result. It doesn't spoil result, as 'summary' function doesn't consider missing values. For example, we can compare average value for 'Food' with 'mean' function.

```
mean(all_spending$Food, na.rm = TRUE)
```

```
## [1] 11.14679
```

The same values. But counting NA values can help us to understand all observations better. So we will keep NA values in our data set.

The 'summary' function is a cool function to get all required statistics and compare values. Looking this data we can immediately draw some conclusions : - on average the highest spending in the category Apartments = about 600, Technics = less than 100, Urgent = about 70 - for other categories, even for Closes, the average speding less than 50 - the most high difference between mean and median is for Home category, but less than 10. The out-lied points will be checked. - we can see that, except 3 highest categories, the median spending for others is in 3-20 - NA values show how often it needs to spending for certain categories : less NA - more often spending and more predictable (except constant spending)

Difference of values (aver and median) allows you to draw certain conclusions. But summary result format is not so good visualized for a lot of columns (like here 12) So it's better to create plots to overview statistical result.

Transform data

To create plot which overviews and compare all categories of spending (Food, Home, ... etc) it needs before to get other format of data frame - long format. Using function pivot_longer to create new data set.

```
all_spending_long <- all_spending %>%
  pivot_longer(names_to='type', values_to = 'expense', Food:Body)
```

```
head(all_spending_long)
```

```
## # A tibble: 6 x 2
##   type     expense
##   <chr>      <dbl>
## 1 Food          60
## 2 Home          80
## 3 Other         NA
## 4 Closes        NA
## 5 Tehcnics     110
## 6 Diime          3
```

It is more convenient to work now with new data set.

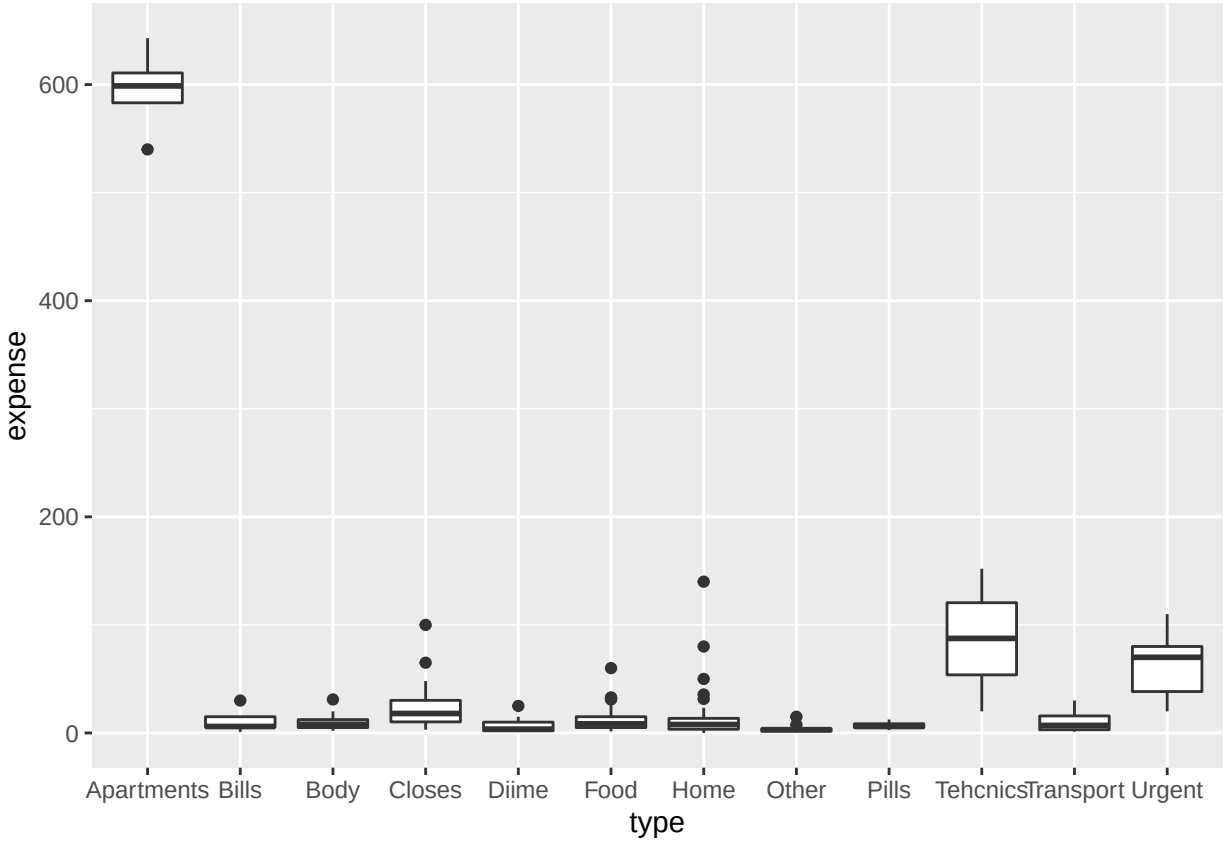
Getting Plots

Using 'ggplot' to get different kind of plots.

As column 'type' of new data set has a categorical values we can apply grouping by type.

First chart: - to know how highway mileage varies across classes by function 'geom_boxplot()'

```
## Warning: Removed 1546 rows containing non-finite values (stat_boxplot).
```



As we can see on chart: - category apartments has only 1 value outlined, with the min distribution, most expensive - the categories like "Technics", "Urgent" and "Transport" don't have outline points, but a little bit diffirent distribution. - the group of rest categories has different outlying points nearby expense = 100, but the distribution is up to expense = 50

We will analyse these groups in more detail later.

Let look trough some another charts which overview all categories.

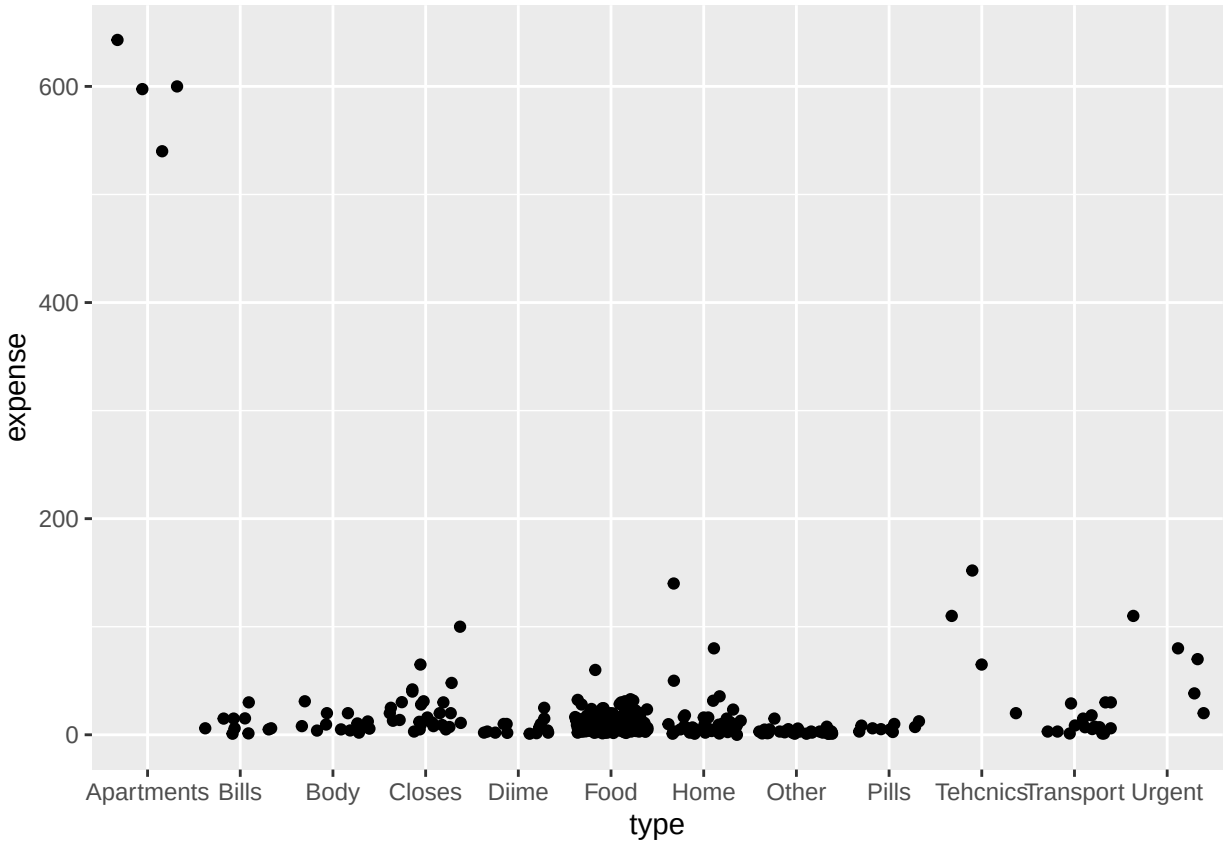
Second chart: - to explore the distribution of a expense values broken down by a categories(expense types)

```
ggplot(data = all_spending_long, mapping = aes(x = expense)) +
  geom_freqpoly(mapping = aes(colour = type), binwidth = 5)
```

As we can see most often spending is for the category 'Food'. The expenses for other categories are the similar for each other. So we look through other plot.

Third chart: - to get the spreading out of expense values, using the jitter plot

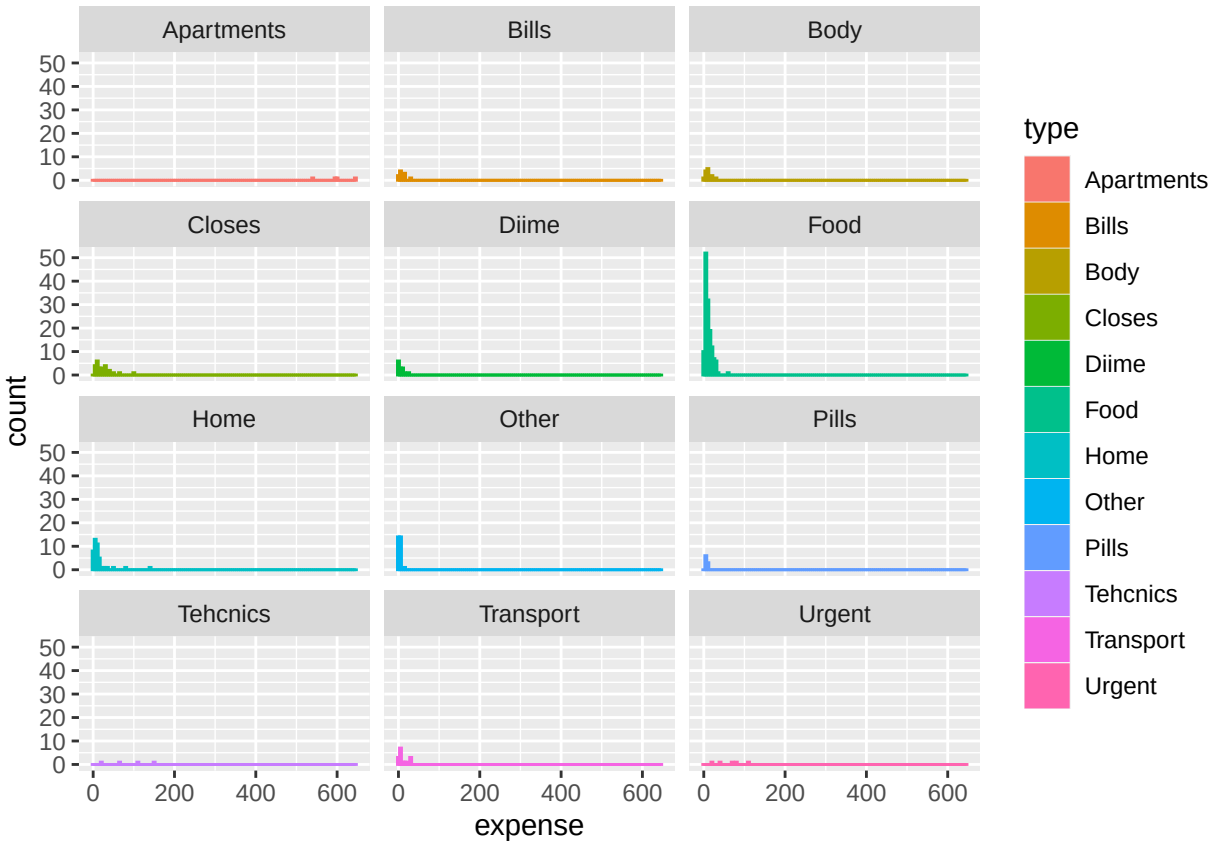
```
## Warning: Removed 1546 rows containing missing values (geom_point).
```



We also can see that many and high point density Food, Home and Closes. Later we will analyse this categories. But before let look to separated chart (histogram) for each category.

Fourth chart: - to get several charts(histogramms), using face_wrap() function

```
## Warning: Removed 1546 rows containing non-finite values (stat_bin).
```



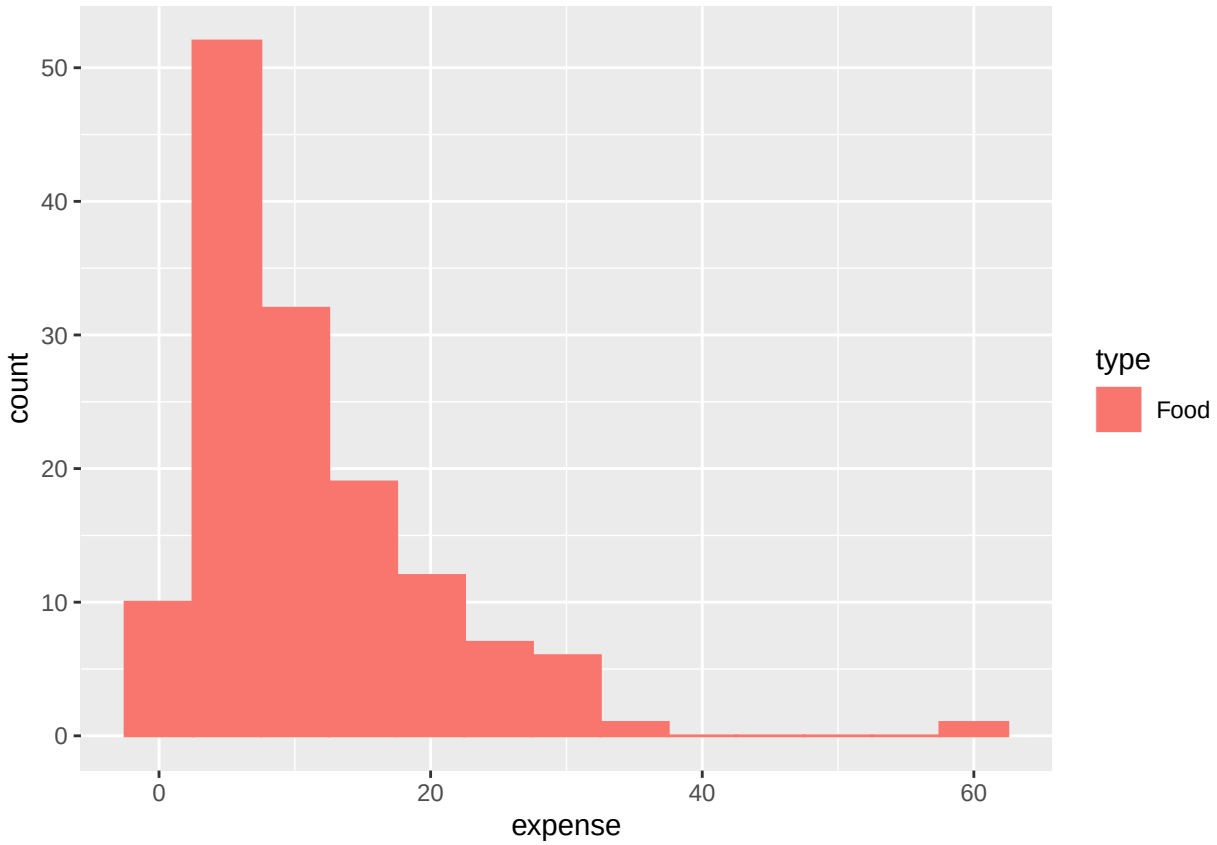
Particular analyses

We have looked through the whole data and get full pictured. Now we can analyses in more detail specific categories.

By category

Let start from Food category which has the high point density. First we create histogram chart just apply filter

```
## Warning: Removed 15 rows containing non-finite values (stat_bin).
```



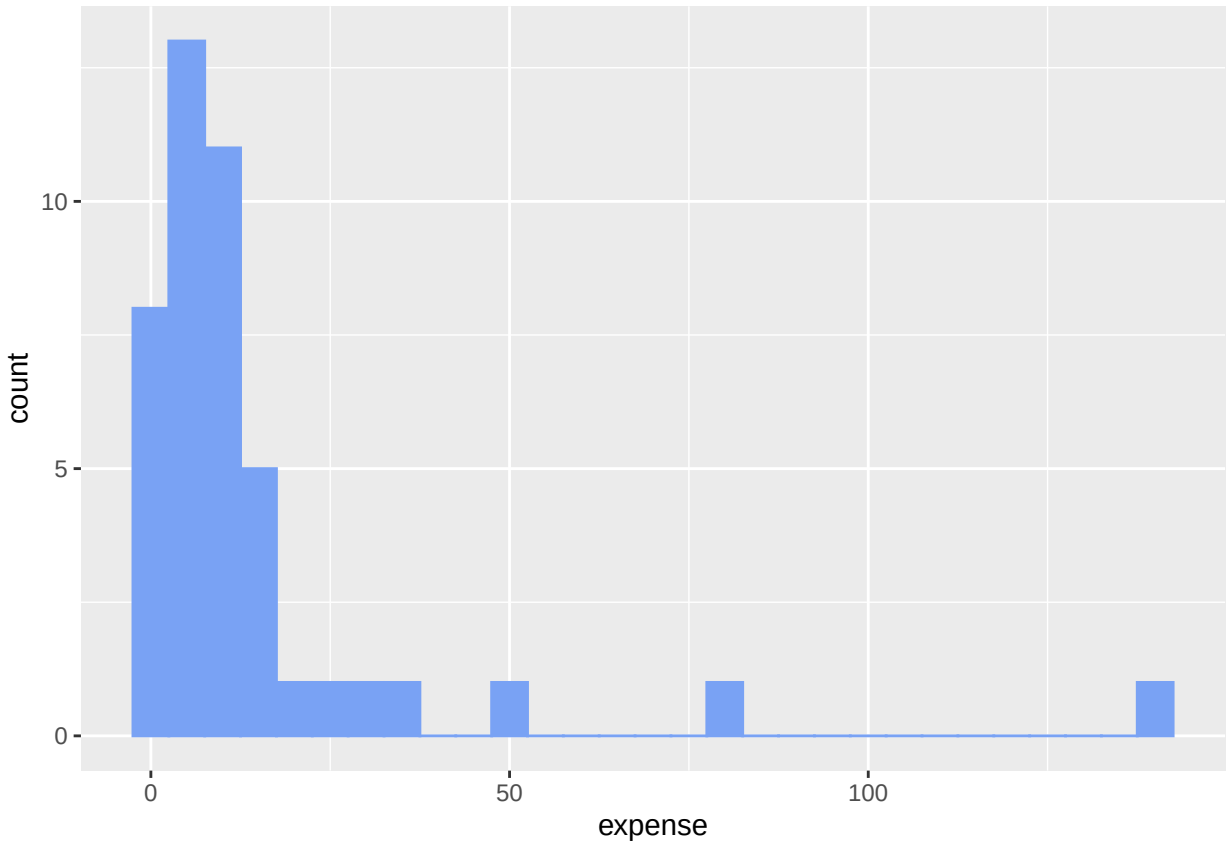
We can see the similar bell-shape histogram and 1 out-lied point. So applying the following characteristics of normal distributions: - 1 times the standard deviation to the left of the mean captures 68.26% of the data - 2 times the standard deviation on either side of the mean captures 95% of the data - 3 times the standard deviation on either side of the mean captures 99.73%

These percentages are true for all data that falls into a normally distributed pattern.

Lets look the histogram for other catagories

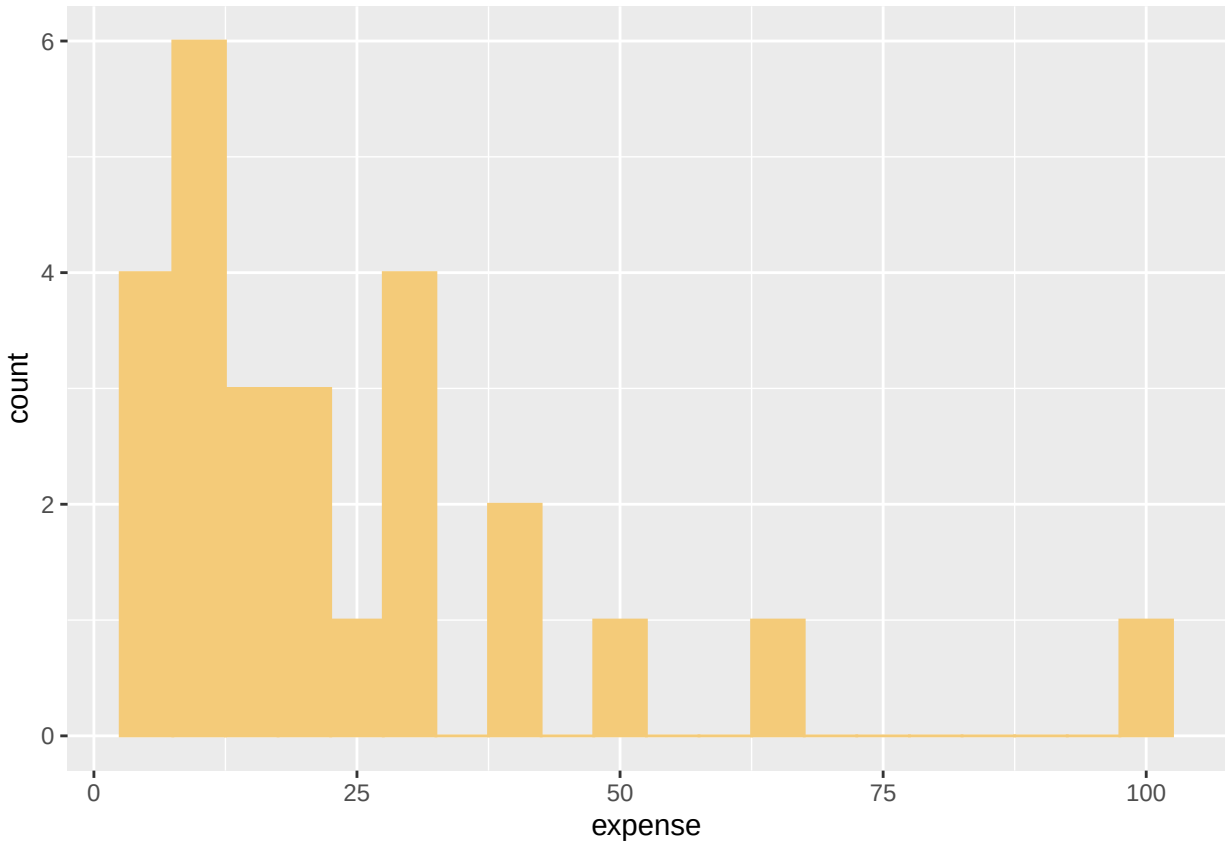
Home There is a quite similar plot to bell-shape histogram, if we consider the outlied like special points. They maybe analyise that what can allows to spend more than 50 and 100, which factors.

```
## Warning: Removed 111 rows containing non-finite values (stat_bin).
```



Closes The plot doesn't look as similar to bell-shape histogram, it seems like to Skewed right form.

```
## Warning: Removed 129 rows containing non-finite values (stat_bin).
```



By month

Let look through the total spending for last full month - November ('Nov' data set).

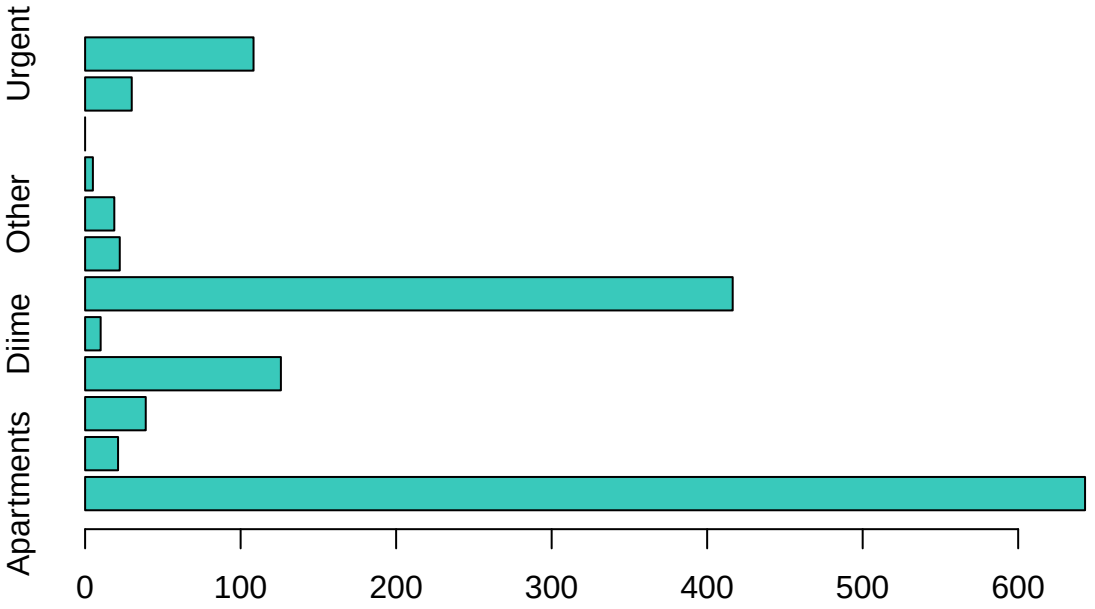
We prepare a data to further analysis (transform to long data)

```
Nov_long <- Nov %>%
  slice_head(n=33) %>%
  pivot_longer(names_to='type', values_to = 'expense', Food:Body)
```

Now we can calculate total spending for each category

```
Nov_stat <- Nov_long %>%
  count(type, wt=expense)

x <- c(Nov_stat$type)
y <- c(Nov_stat$n)
barplot(y, names.arg = x, col = "#39c9bb", horiz = TRUE)
```



Predictions

It's interesting and important to investigate if spending frequency for certain category will explore. To do it we look through NA values and makes some predictions

Prepare data set

First create a new data set: - NA of expense is 0 - no_spend (new colomn) is 1, if expense is NA

```
data_long_nona = all_spending_long %>%
  mutate(no_spend = ifelse(is.na(expense), 1, 0),
         expense = ifelse(is.na(expense), 0, expense))
```

Transform predictions

We can fit a model to it, and generate predictions:

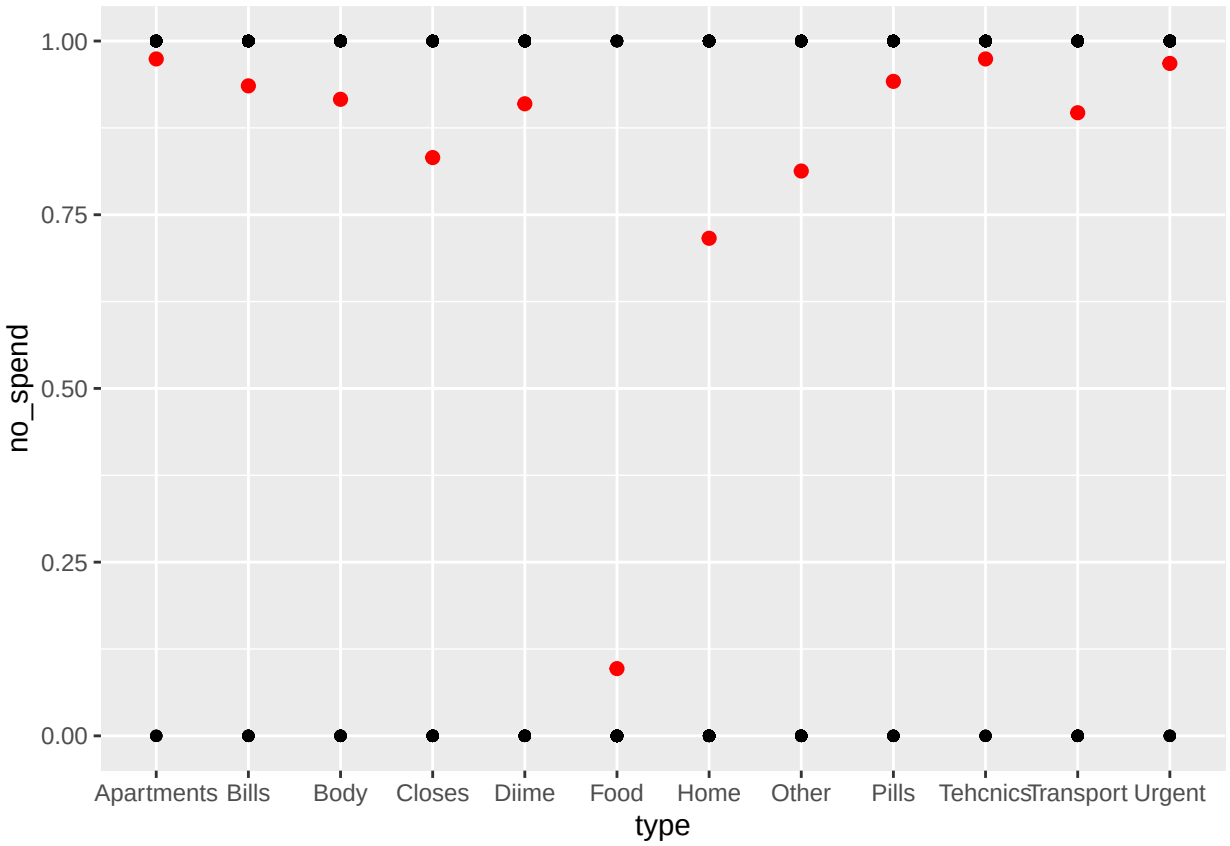
```
  mod2 <- lm(no_spend ~ type, data = data_long_nona)

  grid <- data_long_nona %>%
    data_grid(type) %>%
    add_predictions(mod2)
```

Create a plot

A model above with a categorical 'type' will predict the mean value for each category: - highlighted points with red color.

```
  ggplot(data_long_nona, aes(type)) +
    geom_point(aes(y = no_spend)) +
    geom_point(data = grid, aes(y = pred), colour = "red", size = 2)
```



Interpretation result

The proximity of the black and red points indicates that the frequency of spending will be the same (1 - rarely, 0 - very often). If the red dot is removed towards zero, then the frequency of spending in this category may be increased.

No increasing spending frequency for categories: - Apartments, Urgent, Technics and Pills

Low increasing: - Bills, Body, Diime

Possible significant increasing: - Closes, Home, Other

The Food category has low tendency to drop spending frequency.

Share findings

As we have performed the analysis and gained some insights into the data, we can provide the key points of result. - Total budget - Possible unplanned spending

Total bugget

Let calculate total budget. First we calculate total budget for last full month - November. Then we will add possible predictions.

```
Nov_total <-
  Nov_stat %>% summarise(spend = sum(n))
Nov_total
```

```
## # A tibble: 1 x 1
##   spend
```

```
##   <dbl>
## 1 1440.
```

Possible unplanned spending

As we can see from EDA analysis some categories don't have outline points (or minumun disctrubution), others - many and high point density. So for the following categories we will apply maximum value, mean(or median), or 3rd Quater value (75 %). Also we consider the prediction result about increasing (or not) of spending frequency.

```
dim(all_spending)
```

```
## [1] 155  12
```

Categries without outline low (or not) increasing of spending frequency points, apply 3rd Quater value (75 %) - Apartments value 610.8 - Urgent 80 - Technics 120.5 - Pills 8.5 - Transport value 15.75 - Bills 15 - Body 12.34 - Diime 10

Categories with high point density, normal distributions or not Normal distribution - apply mean value - Food 11.5

Possible significant increasing similar to the Normal distribution - apply mean - Home 14.2 not looks like to normal distribution, apply 3rd Quarter value (75 %) - Closes 100 - Others 15

Each value above is per 1 observation. As we can know how many observations we have, we can predict the possible spending (unplanned). dim(all_spending) Total observation is 155 for all month (4.5 month).

Due to NA counts in each category and spending frequency (increasing or not), we can predict bugget per month - Apartments value 610.8 * 1 per month - Urgent 80 * 1 per month - Technics 120.5 * 1 per month - Pills 8.5 * 2 per month - Transport value 15.75 * 3.5 per month - Bills 15 * 2 per month - Body 12.34 * 2.9 per month - Diime 10 * 2.9 per month

- Food 11.5 * 31 per month
- Home value 14.2 * 12.2 (plus 25 % increase)
- Closes 30.15 * 6.6(15 % increase)
- Others 4 * 7.4 (15 % increase)

Total predict budget - 1736.54