Data Science: Capstone Chose Your Own Projct Report

Louri Compain

2023-12-03

Contents

1 Overview

1.1 Introduction

This report is my submission for the "Project Submission: Choose Your Own" part of the HarvardX PH125.9x Data Science: Capstone course.

I elected to use a dataset from kaggle, titled "OCD Patient Dataset: Demographics & Clinical Data", with information about 1500 patients. This is a medical dataset that can be used for the very practical purpose of helping a patient with an OCD diagnosis. Please take note that I am not a medical professional and the purpose of this project is not a deployment in a professional context. The goal here is to practice my data science skills and to showcase it for the class.

1.2 Project goal

My goal during this project will be to predict the Duration of Symptoms variable, based on the rest of the information available about the patient

This is a regression task. We can note that this is a "true regression" as we have a continuous range of possible values, unlike the MovieLens exercise, where I used regression and rounded to set values to predict an ordinal class.

To measure the performance of our predictions, we will be using the RMSE.

1.2.1 Libraries and Dataset

```
#------libraries loading--------------
library(readr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(caret)
```

```
## Le chargement a nécessité le package : lattice
##
## Attachement du package : 'caret'
##
## L'objet suivant est masqué depuis 'package:purrr':
##
##     lift
```

```
library(e1071)
library(ggplot2)

library(corrplot)
```

```
## Warning: le package 'corrplot' a été compilé avec la version R 4.3.2
```

```
## corrplot 0.92 loaded
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attachement du package : 'randomForest'
##
## L'objet suivant est masqué depuis 'package:dplyr':
##
##     combine
##
## L'objet suivant est masqué depuis 'package:ggplot2':
##
##     margin
```

```
library(h2o)
```

```
##
## ----------------------------------------------------------------------
##
## Your next step is to start H2O:
##     > h2o.init()
##
## For H2O package documentation, ask for help:
##     > ??h2o
##
## After starting H2O, you can use the Web UI at http://localhost:54321
## For more information visit https://docs.h2o.ai
##
## ----------------------------------------------------------------------
##
##
## Attachement du package : 'h2o'
##
## Les objets suivants sont masqués depuis 'package:lubridate':
##
##     day, hour, month, week, year
##
## Les objets suivants sont masqués depuis 'package:stats':
##
##     cor, sd, var
##
## Les objets suivants sont masqués depuis 'package:base':
##
##     %*%, %in%, &&, ||, apply, as.factor, as.numeric, colnames,
##     colnames<-, ifelse, is.character, is.factor, is.numeric, log,
##     log10, log1p, log2, round, signif, trunc
```

```
#------seed setting--------------
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
#------dataset--------------
library(here)
```

```
## Warning: le package 'here' a été compilé avec la version R 4.3.2
```

```
## here() starts at C:/Users/Admin/Desktop/Business/projets/mooc pro/EDX/HarvardX Capstone/workspace 2
```

```
here("ocd_patient_dataset.csv")
```

```
## [1] "C:/Users/Admin/Desktop/Business/projets/mooc pro/EDX/HarvardX Capstone/workspace 2/ocd_patient_
```

```
ocd_patient_dataset <- read_csv(here("ocd_patient_dataset.csv"))
```

```
## Rows: 1500 Columns: 17
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (11): Gender, Ethnicity, Marital Status, Education Level, Previous Diag...
## dbl   (5): Patient ID, Age, Duration of Symptoms (months), Y-BOCS Score (Obs...
## date  (1): OCD Diagnosis Date
##
## 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.
```

We have 31 variables: - Patient ID: a numerical Id to recognize a patient while protecting their privacy. It does not reveal information about them, and we can expect each patient to apear only once, so its usefullness is limited. - Age: Age of the patient at the time of data collection. Interval variable in years. - Gender: Gender of the patient. Nominal variable inthe form of a binary between men and women. - Ethnicity: Ethnic background of the patient. Nominal variable with the following values: African, Asian, Caucasion, Hispanic. - Marital Status: Marital status of the patient. Nominal variable. The possible values are Single, Married or Divorced. - Education Level: Highest level of education completed by the patient. Ordinal variables with the following values: High School, Some College, College Degree, Graduate Degree. - OCD Diagnosis Date : Date when the patient was diagnosed with Obsessive-Compulsive Disorder.This is an interval variable in days, with values between 13/11/2013 and09/11/2022. - Duration of Symptoms (months): Number of months since the onset of OCD symptoms. Interval variable between 6 and 240. This is the variable we want to predict for the project. - Previous Diagnoses: Any previous psychiatric diagnoses or comorbidities. This nominal variable has values : GAD, MDD, None, Panic Disorder, PTSD. - Family History of OCD: Indicates whether there is a family history of OCD in the patient's relatives. This is a binary categorical variable. - Obsession Type: Type of obsession (no description on the dataset page). This is a nominal variable. The values are Contamination, Harm-Related, Hoarding, Religious, Symmetry. - Compulsion Type: Type of compulsion (no description on the dataset page). This is a nominal variable. The values are Checking, Counting, Ordering, Praying and Washing. - Y-BOCS Score (Obsessions): Score on the Yale–Brown Obsessive Compulsive Scale for Obsessions(no description on the dataset page). Interval variable between 0 and 40. - Y-BOCS Score (Compulsions): Score on the Yale–Brown Obsessive Compulsive Scale for Compulsions(no description on the dataset page). Interval variable between 0 and 40. - Depression Diagnosis: Diagnosis for depression as a binary variable (no description on the dataset page). - Anxiety Diagnosis: Diagnosis for anxiety as a binary variable (no description on the dataset page). - Medications: Medication given to the patient (no description on the dataset page). This is a nominal variable with the following values : Benzodiazepine, None, SNRI, SSRI.

```
head(ocd_patient_dataset)
```

```
## # A tibble: 6 x 17
##   `Patient ID`   Age Gender Ethnicity `Marital Status` `Education Level`
##          <dbl> <dbl> <chr>  <chr>     <chr>            <chr>
## 1         1018    32 Female African   Single           Some College
## 2         2406    69 Male   African   Divorced         Some College
```

```
## 3         1188    57 Male   Hispanic  Divorced         College Degree
## 4         6200    27 Female Hispanic  Married          College Degree
## 5         5824    56 Female Hispanic  Married          High School
## 6         6946    32 Female Asian     Married          College Degree
## # i 11 more variables: `OCD Diagnosis Date` <date>,
## #   `Duration of Symptoms (months)` <dbl>, `Previous Diagnoses` <chr>,
## #   `Family History of OCD` <chr>, `Obsession Type` <chr>,
## #   `Compulsion Type` <chr>, `Y-BOCS Score (Obsessions)` <dbl>,
## #   `Y-BOCS Score (Compulsions)` <dbl>, `Depression Diagnosis` <chr>,
## #   `Anxiety Diagnosis` <chr>, Medications <chr>
```

We will immediately deal with one issue of this dataset: several columns have multi-word names, which might not react well to some functions. We can change that by renaming them.

```
ocd_patient_dataset <- as.data.frame(ocd_patient_dataset)

names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Patient ID"] <- "Patient_ID"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Marital Status"] <- "Marital_Status"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Education Level"] <- "Education_Level"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "OCD Diagnosis Date"] <- "OCD_Diagnosis_Date"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Duration of Symptoms (months)"] <- "Duration_
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Previous Diagnoses"] <- "Previous_Diagnoses"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Family History of OCD"] <- "Family_History_of_
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Obsession Type"] <- "Obsession_Type"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Compulsion Type"] <- "Compulsion_Type"
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Y-BOCS Score (Obsessions)"] <- "Y_BOCS_Score_
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Y-BOCS Score (Compulsions)"] <- "Y_BOCS_Score_
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Depression Diagnosis"] <- "Depression_Diagnosi
names(ocd_patient_dataset)[names(ocd_patient_dataset) == "Anxiety Diagnosis"] <- "Anxiety_Diagnosis"
```

We will now separate the dataset between a main working set used for exploration and training, and a separate set for the final test. We do not have many data points, so a rather small test set is preferable. A 10% separation should give us 1350 data points for the work set and 150 for the test set. This should give us enough in the test set to have something to measure, while preserving many points for the training.

```
test_index <- createDataPartition(y = ocd_patient_dataset$`Duration_of_Symptoms`, times = 1, p = 0.1, li
work_set <- ocd_patient_dataset[-test_index,]
final_test_set <- ocd_patient_dataset[test_index,]
```

##Report plan We will be

2 Analysis

2.1 Initial Analysis

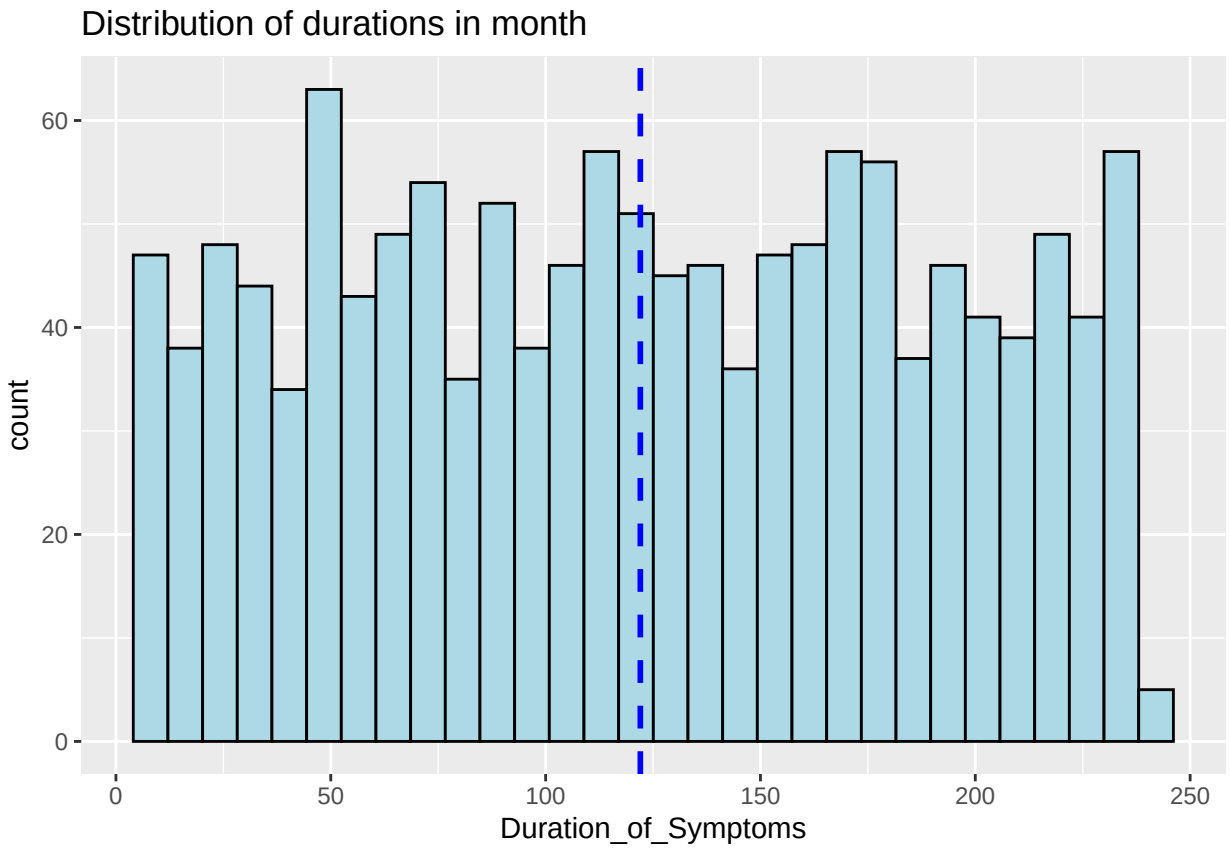
Before we make more changes to the dataset, we will be giving a look to some of the values. ###Predicted variable distribution The first we can do is see the distribution of the variable we want to predict.

```
ggplot(work_set, aes(x=`Duration_of_Symptoms`)) +
  geom_histogram(color="black", fill="lightblue") +
  ggtitle("Distribution of durations in month")+
  geom_vline(
    aes(
      xintercept=mean(Duration_of_Symptoms)
      ),
    color="blue",
```

```
    linetype="dashed",
    size=1
  )
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We have values ranging from 6 to 240, with an average of 122.0504. This is a value in months.

Now, we will consider the variables available for the prediction

2.1.1 ID

```
n_distinct(work_set$Patient_ID)
```

```
## [1] 1260
```

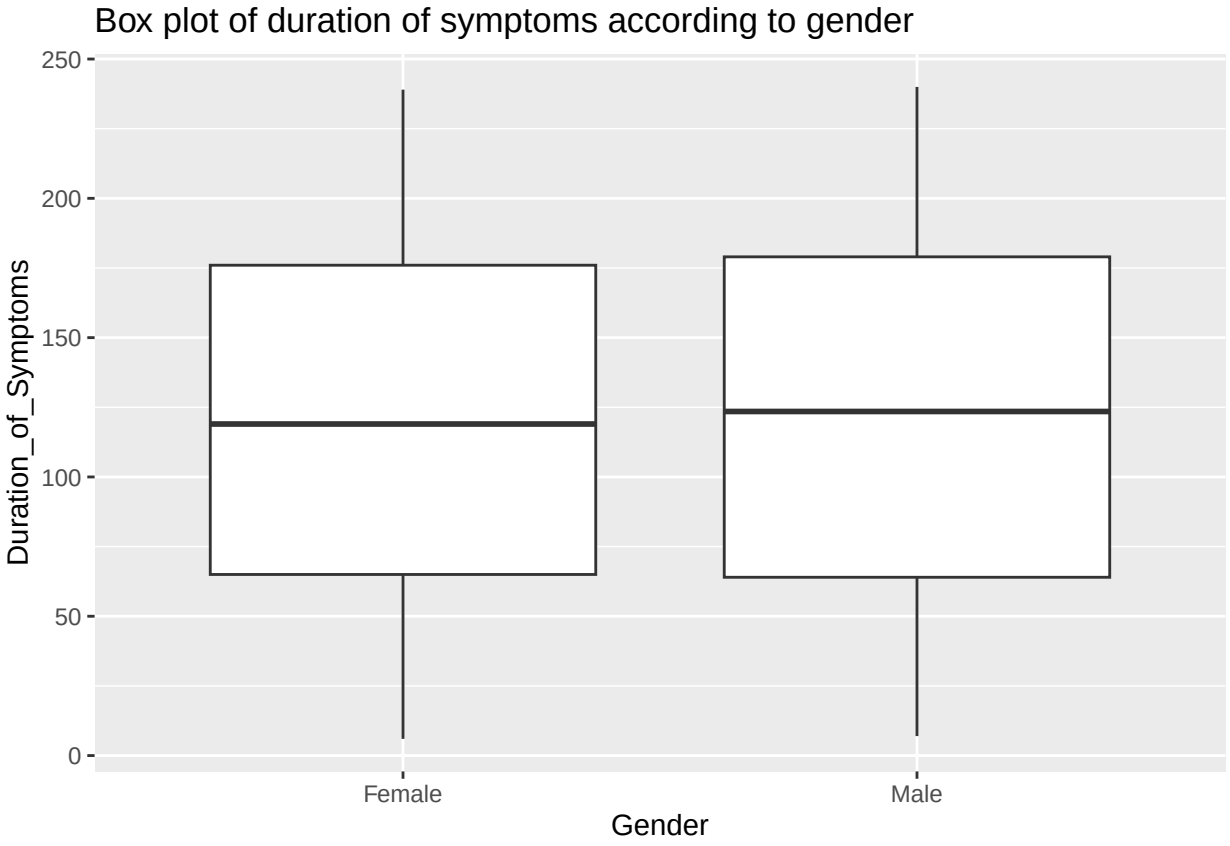
```
nrow(work_set)
```

```
## [1] 1349
```

We can see that 87 patients have two entries. This is not really worth exploiting for our project, since we cannot know for sure that a patient will have previous history (like we did in the MovieLens project for users).

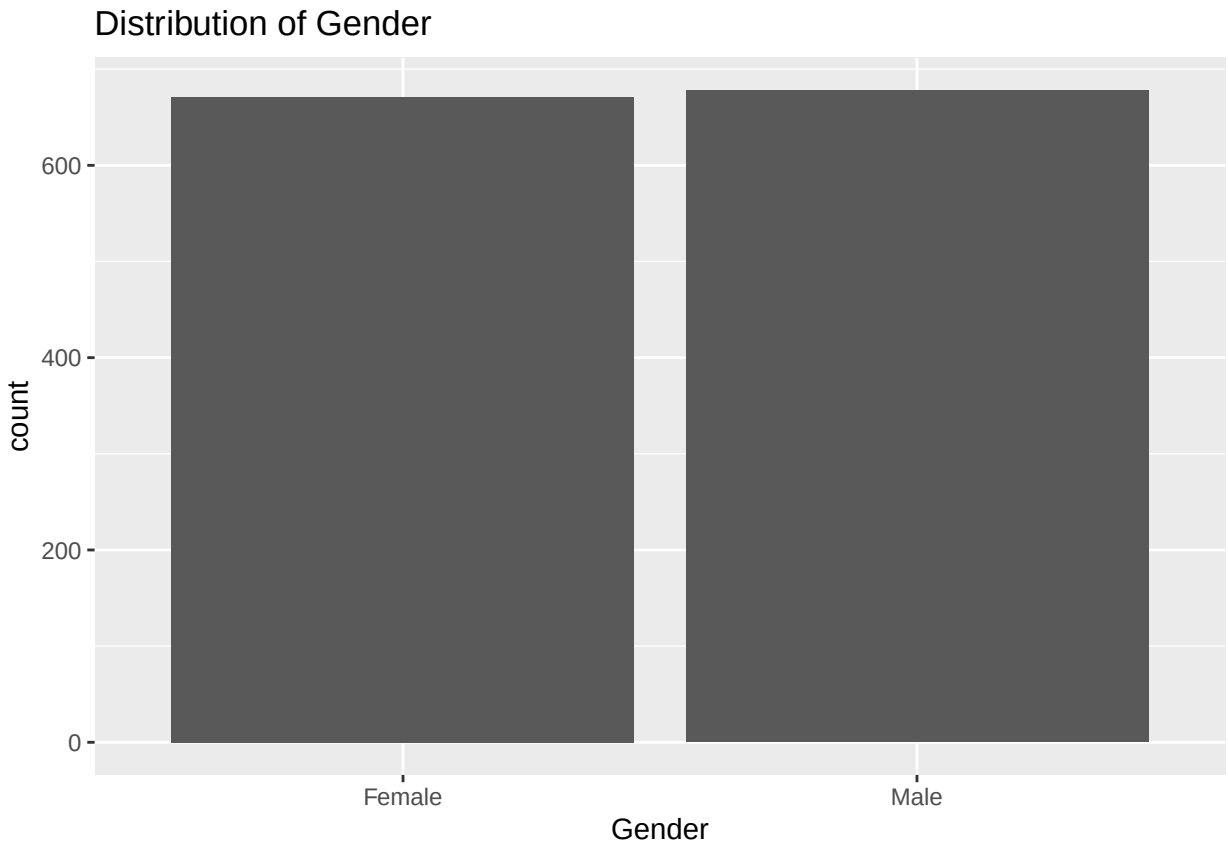
2.1.2 Gender

```
work_set %>% ggplot(aes(x=Gender, y=`Duration_of_Symptoms`)) + geom_boxplot()  + ggtitle("Box plot of du
```



We can see that men have a slightly higher dispersion.

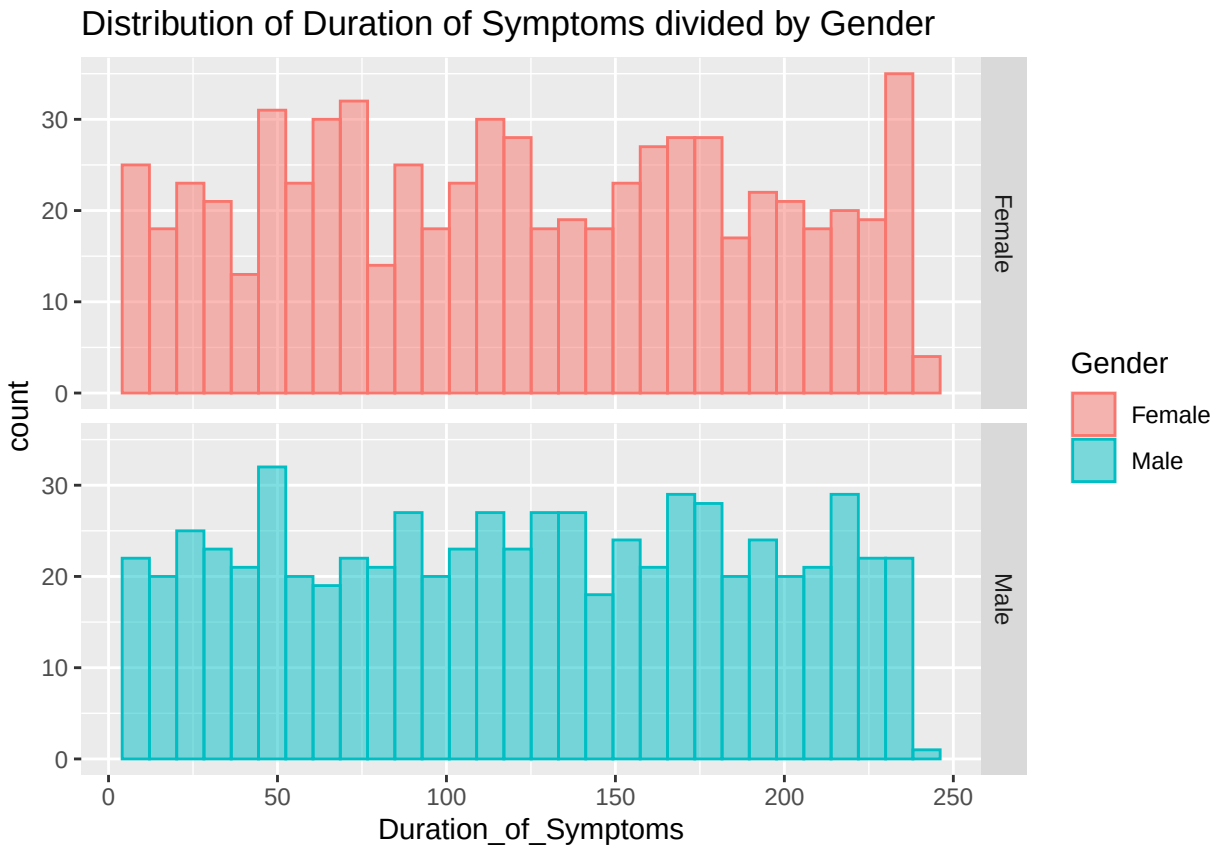
```
ggplot(work_set, aes(x=`Gender`)) + geom_bar() + ggtitle("Distribution of Gender")
```



Perfect parity is nice, and reduces the risk of having a tool that predicts better for one gender than the other, but is also not very informative.

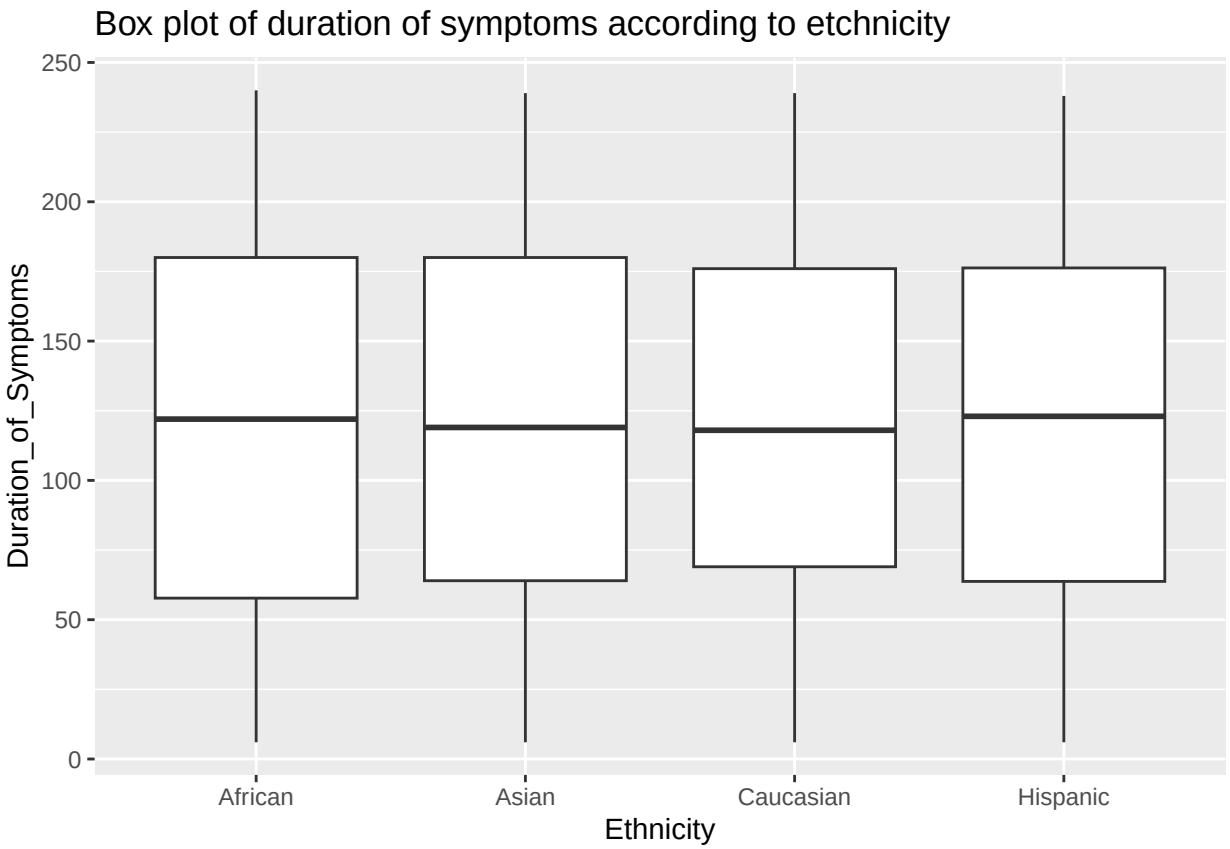
```
ggplot(work_set, aes(x=Duration_of_Symptoms, fill=Gender, color=Gender)) +
  geom_histogram(alpha=0.5, position="identity")+
  facet_grid(Gender ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Gender")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



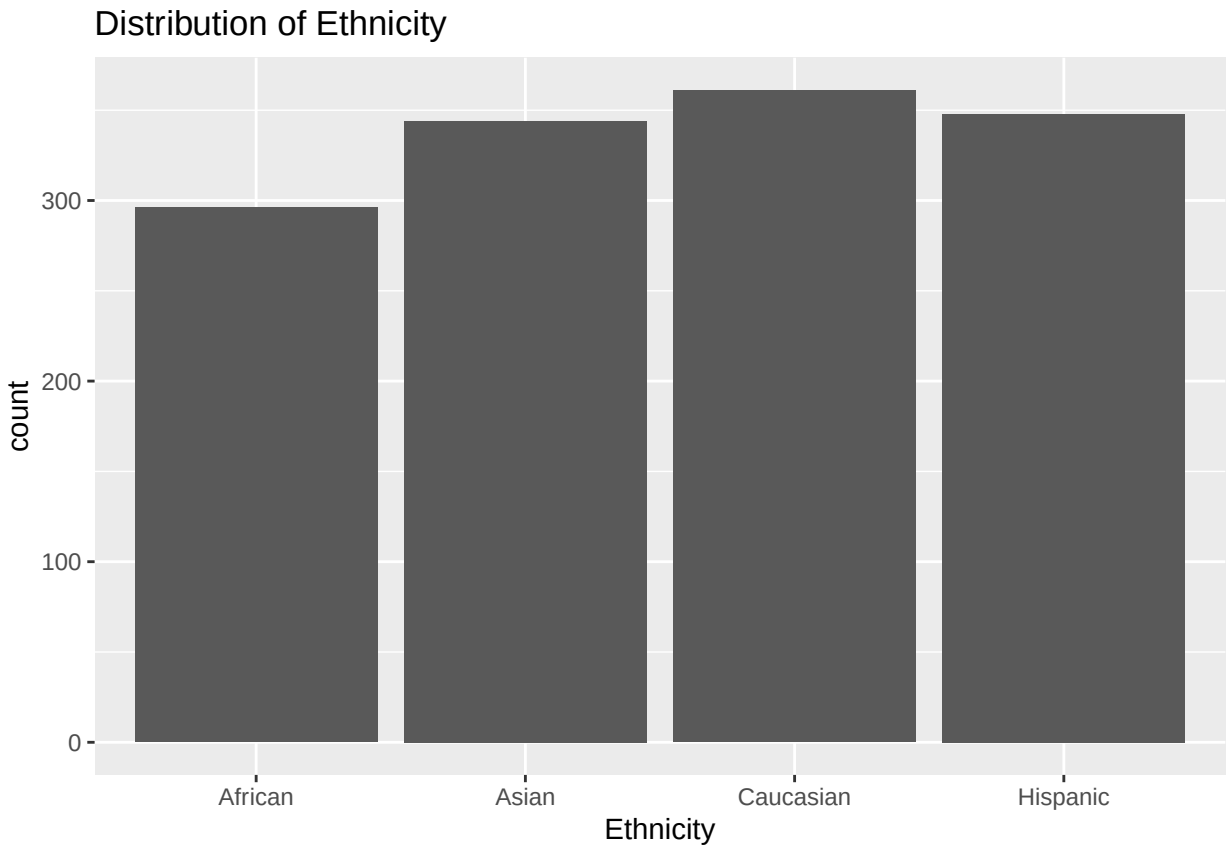
2.1.3 Ethnicit

```
work_set %>% ggplot(aes(x=Ethnicity, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to etchnicity")
```



We can see some change between ethnicities.

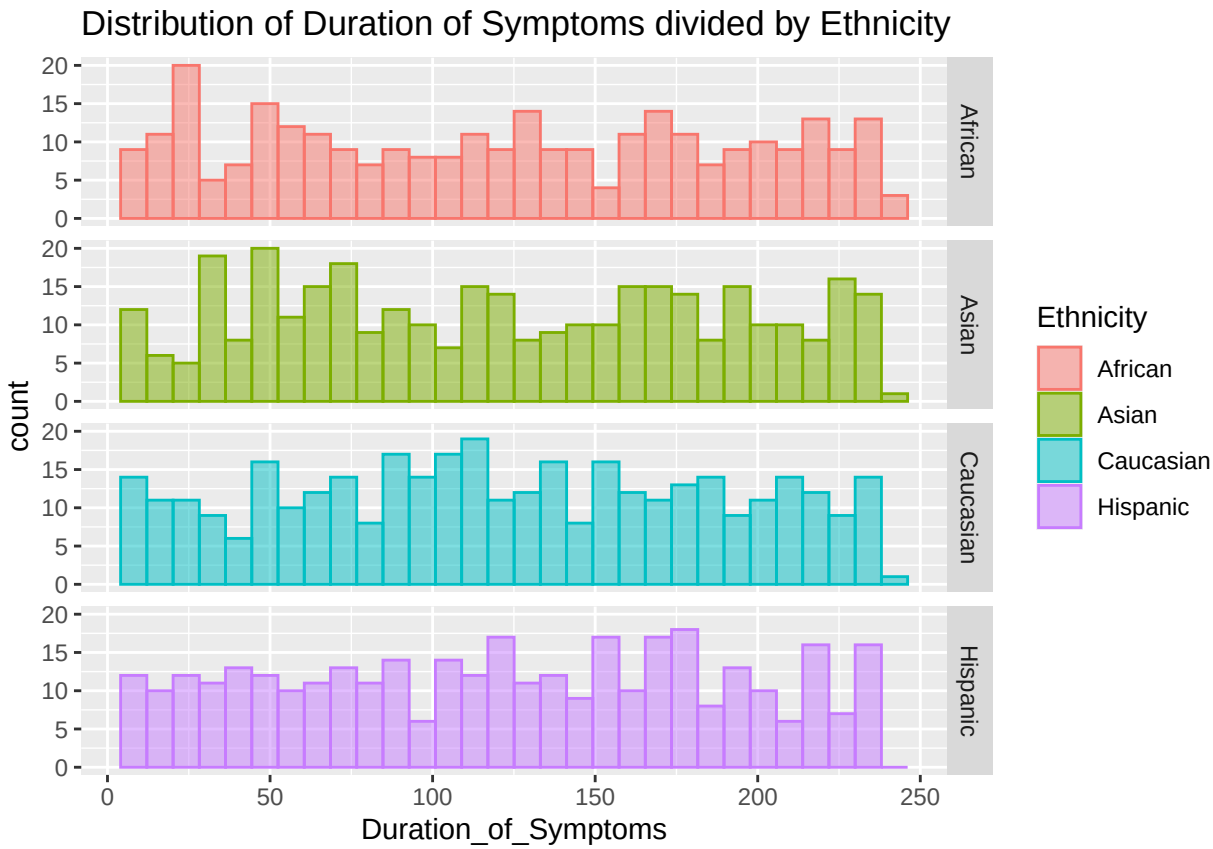
```
ggplot(work_set, aes(x=`Ethnicity`)) + geom_bar() + ggtitle("Distribution of Ethnicity")
```



Our distribution of ethnicities is inequal, but not radically.

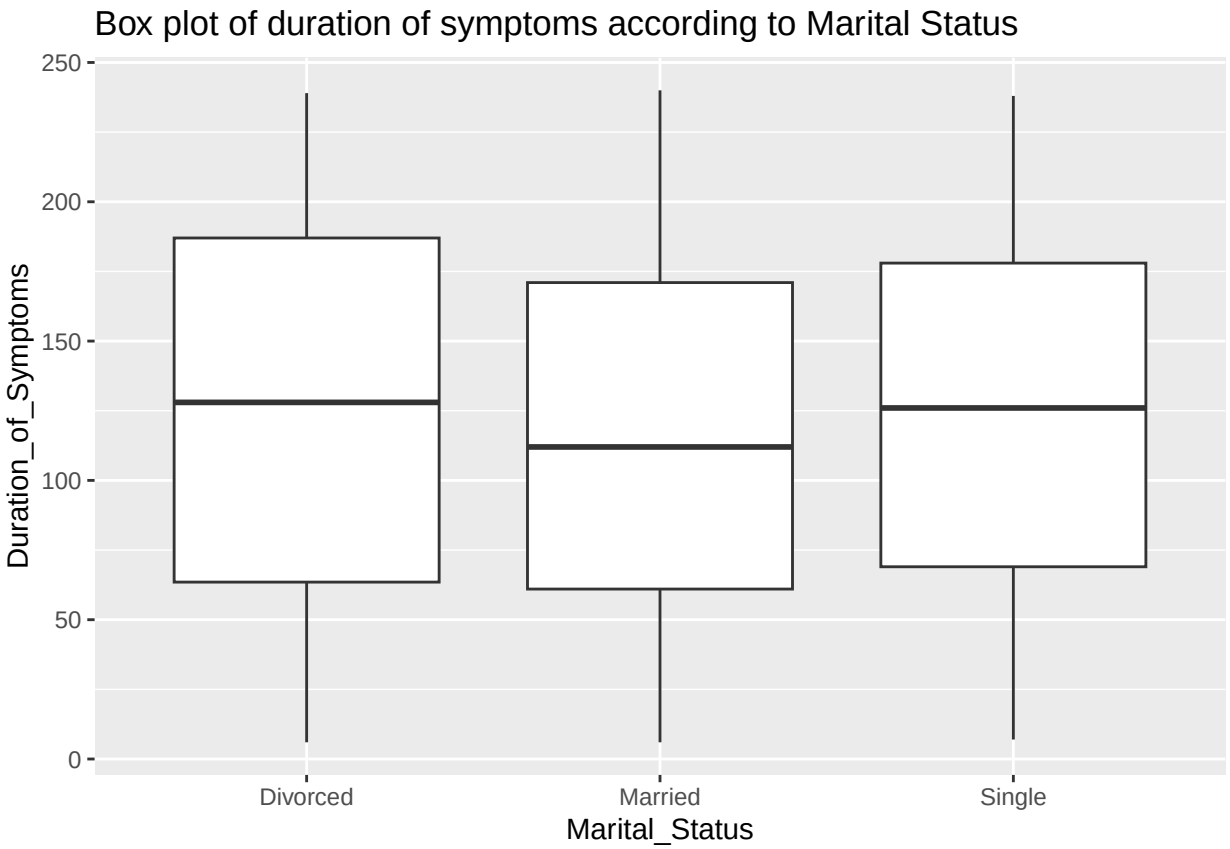
```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Ethnicity, fill=Ethnicity)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Ethnicity ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Ethnicity")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



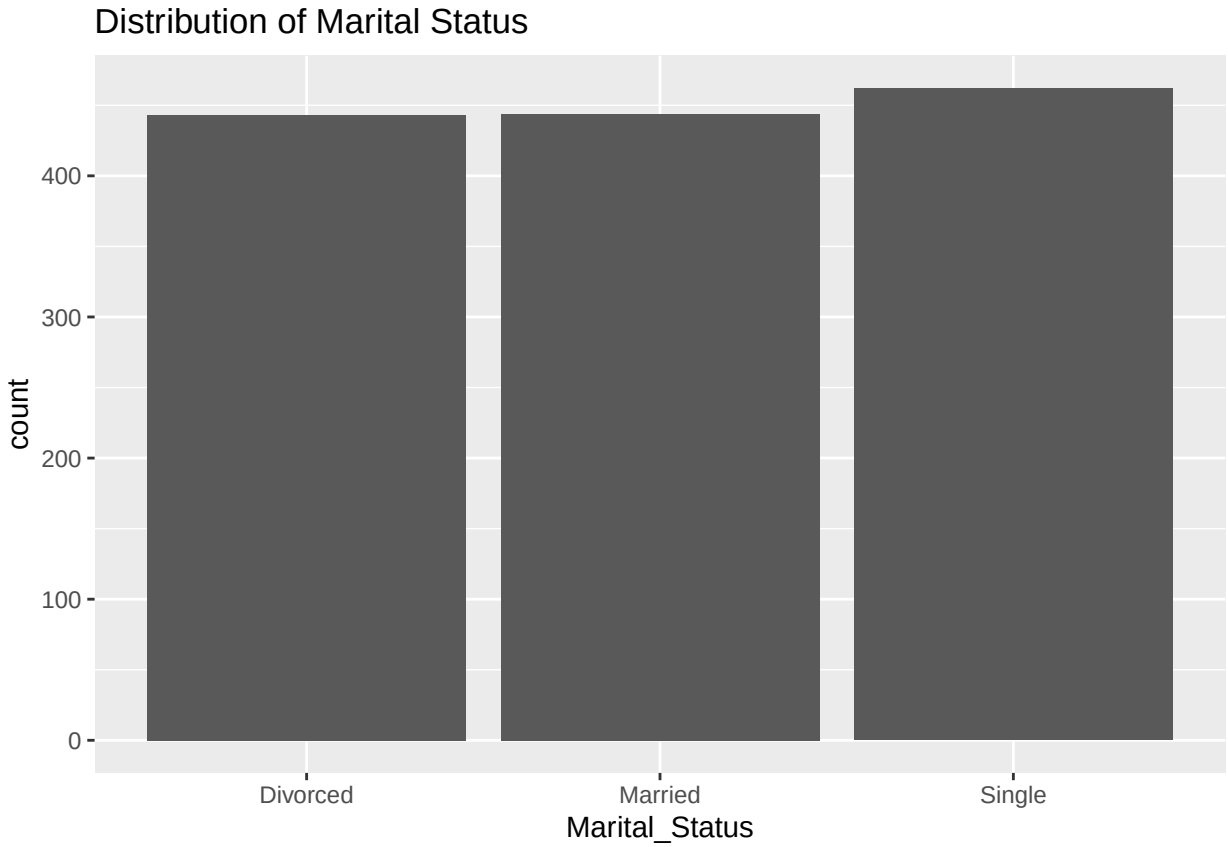
2.1.4 Marital Status

```
work_set %>% ggplot(aes(x=`Marital_Status`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Marital Status")
```



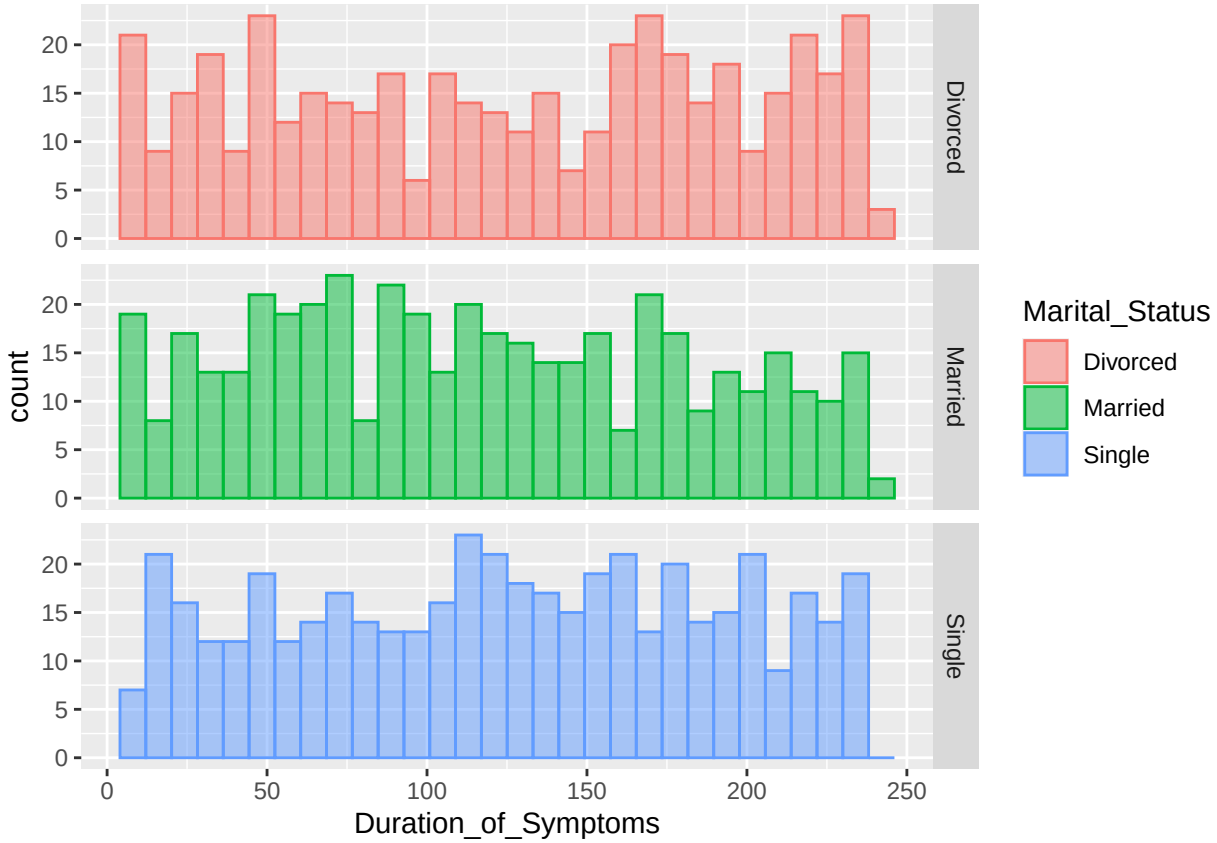
Maried people seem to fare a bit better, with lowe average and a smaller box than divorced or single people.

```
ggplot(work_set, aes(x=`Marital_Status`)) + geom_bar() + ggtitle("Distribution of Marital Status")
```



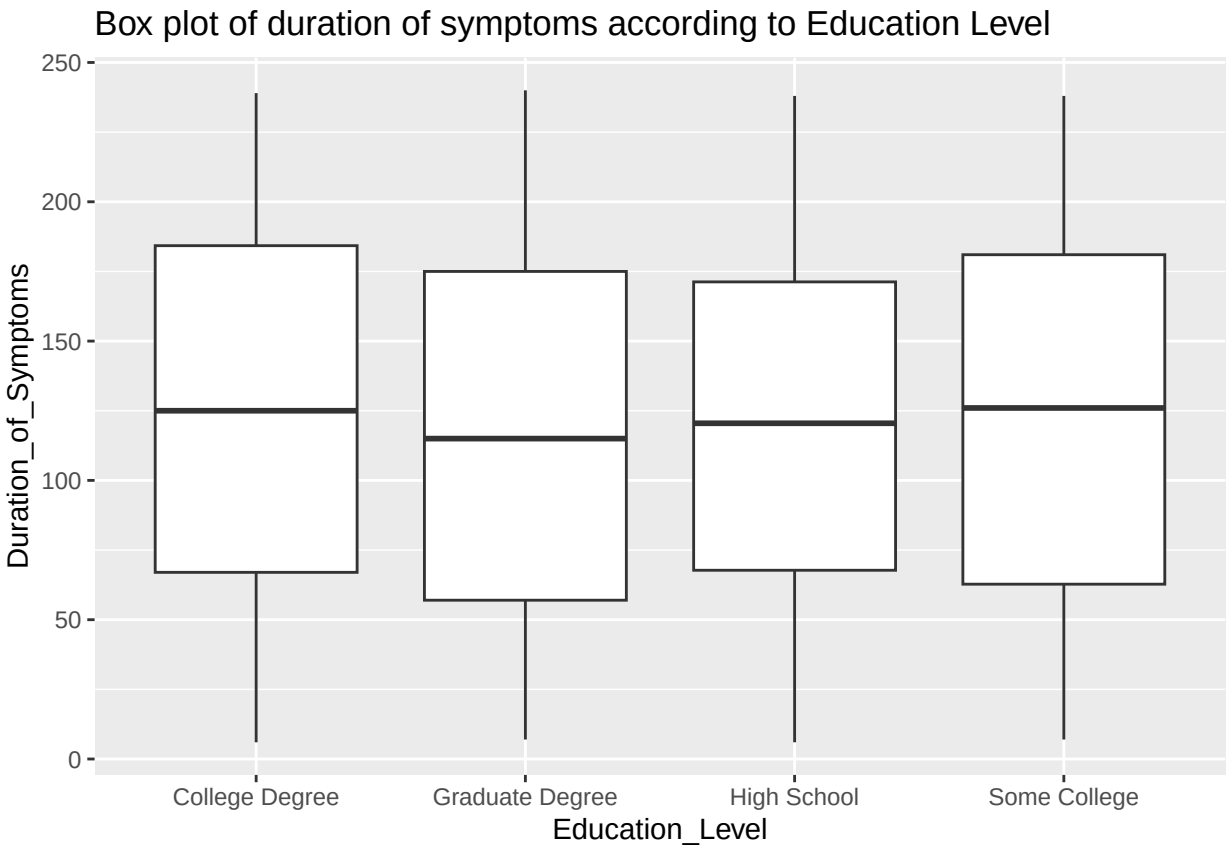

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Marital_Status, fill=Marital_Status)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Marital_Status ~ .)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



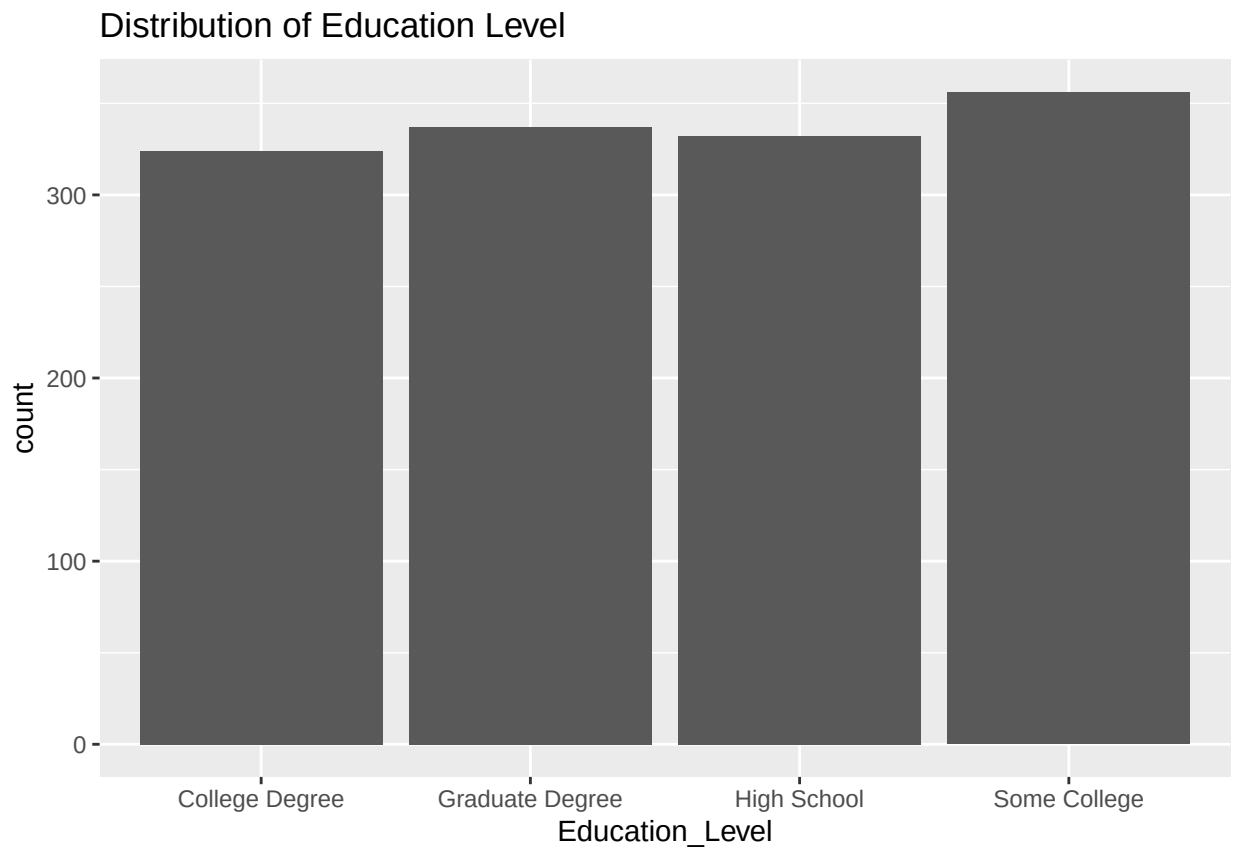
2.1.5 Education Level

```
work_set %>% ggplot(aes(x=`Education_Level`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Education Level")
```



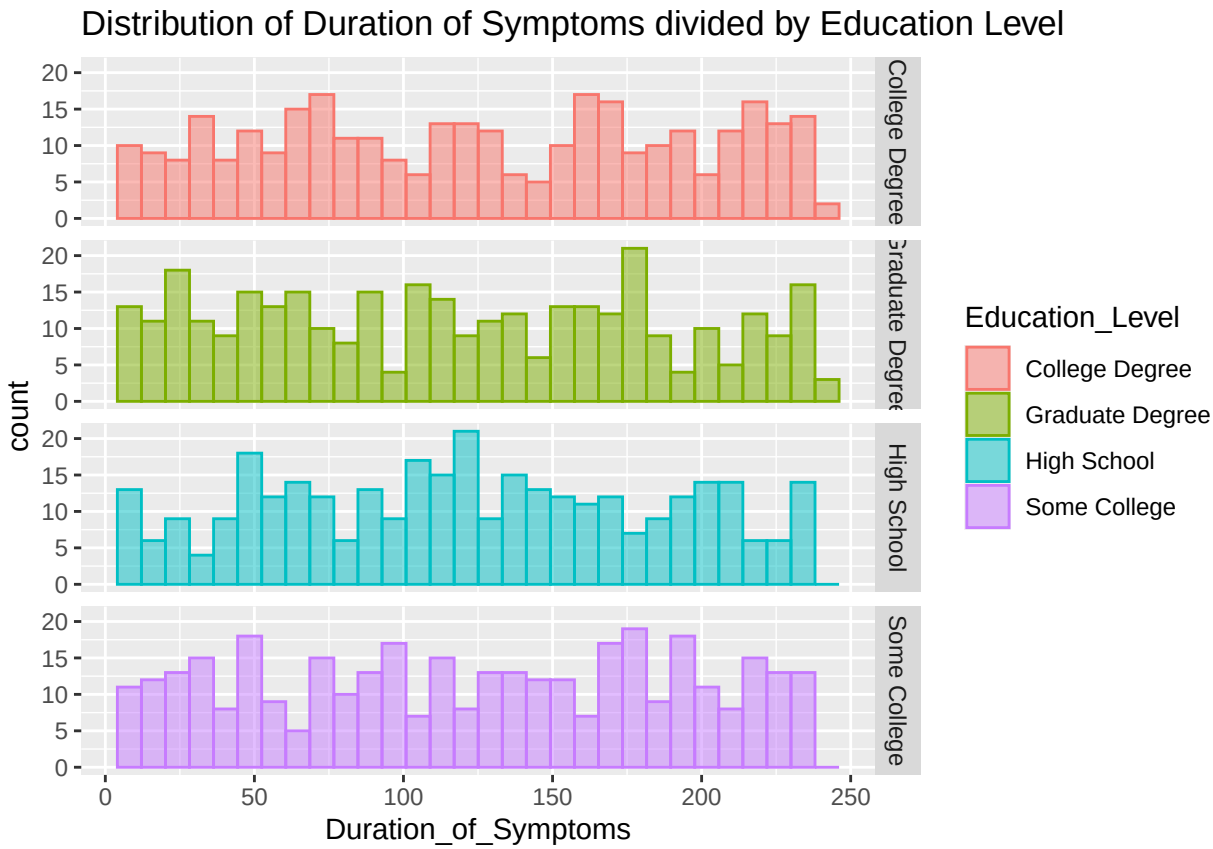
We see some variation, and in particular for the Graduate Degree with a visibly lower average and lower bounds for te box.

```
ggplot(work_set, aes(x=`Education_Level`)) +
  geom_bar() +
  ggtitle("Distribution of Education Level")
```



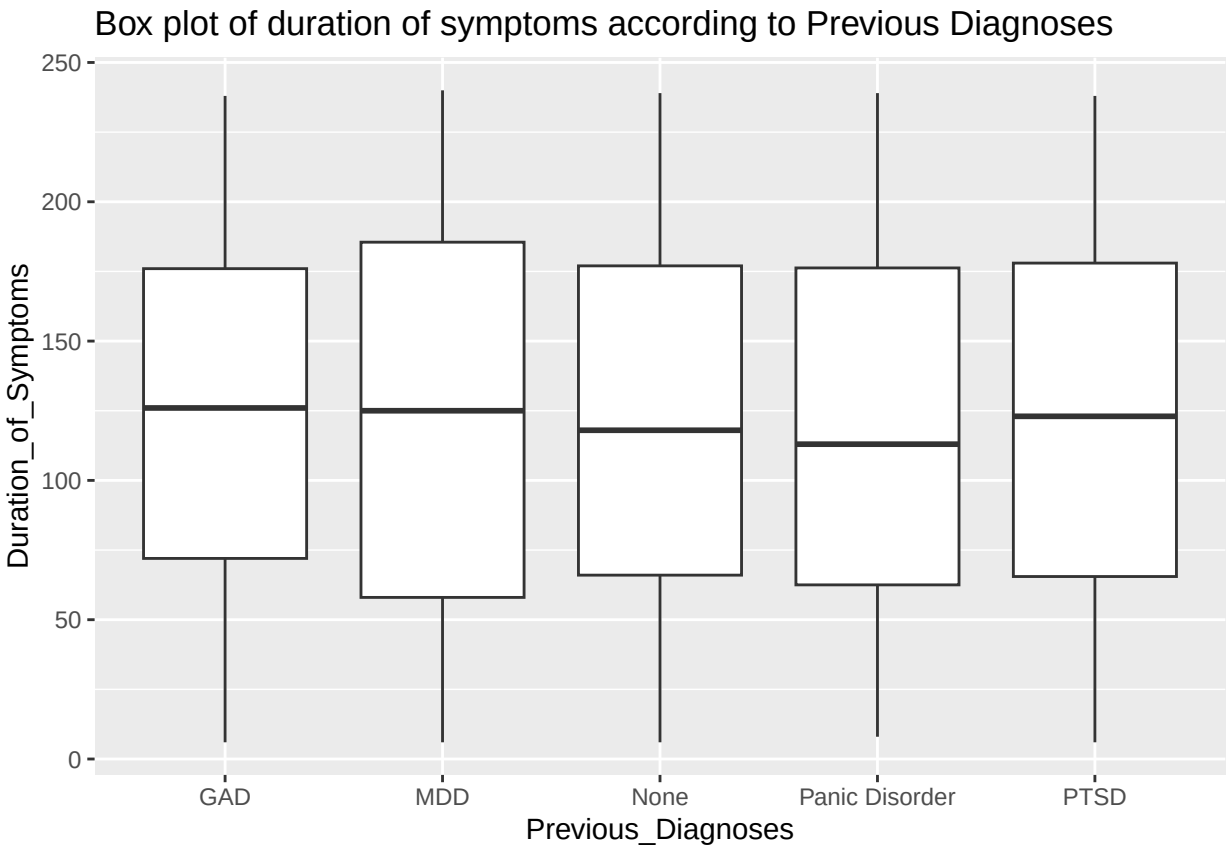

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Education_Level, fill=Education_Level)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Education_Level ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Education Level")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

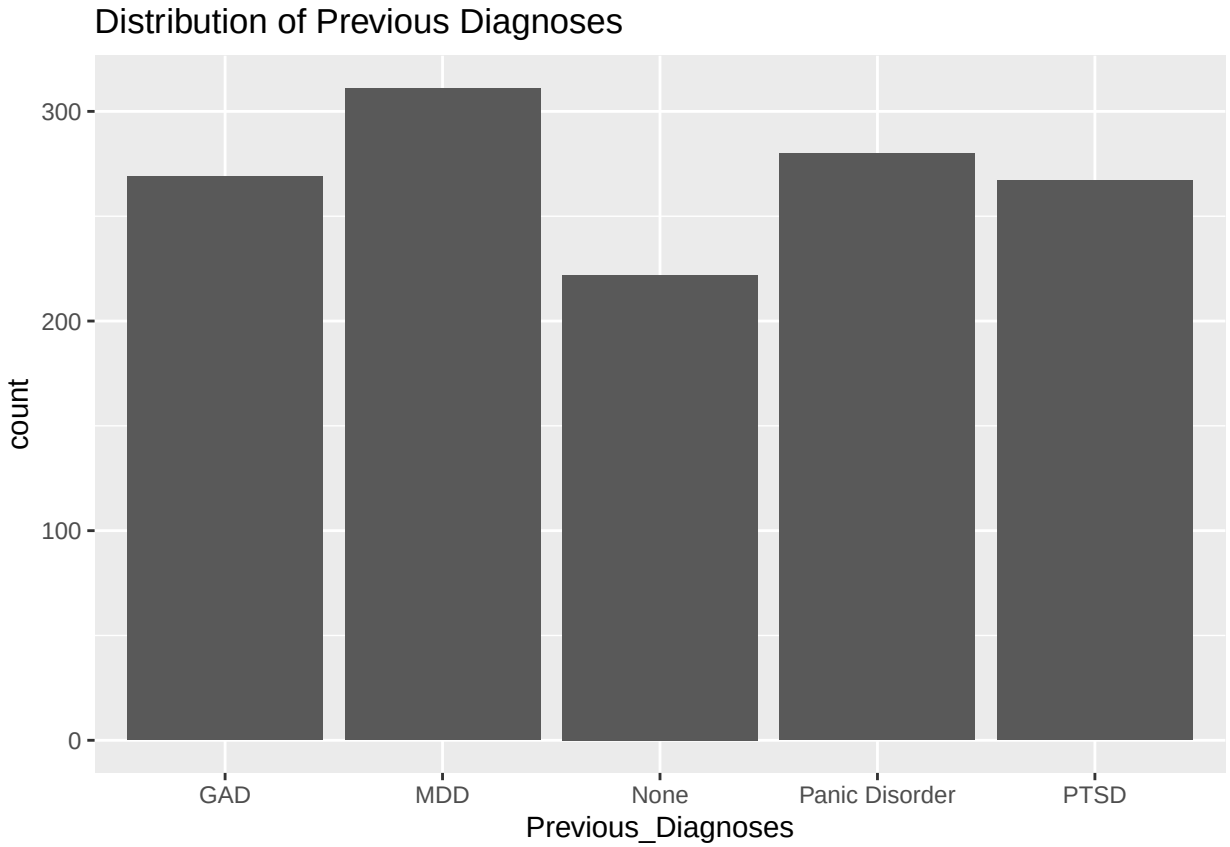


2.1.6 Previous Diagnoses

```
work_set %>% ggplot(aes(x=`Previous_Diagnoses`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Previous Diagnoses")
```



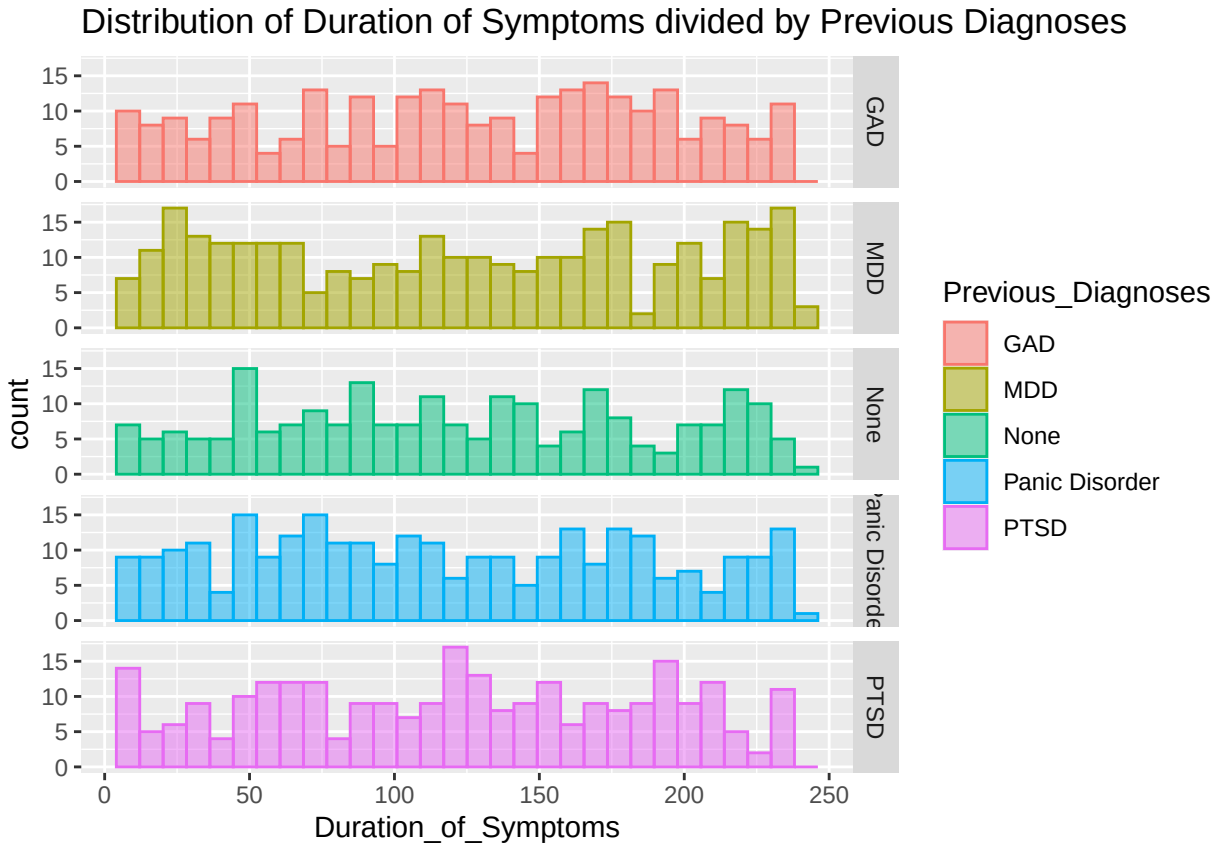

```
ggplot(work_set, aes(x=`Previous_Diagnoses`)) + geom_bar() + ggtitle("Distribution of Previous Diagnoses
```



We can notice that None is the rarest value.

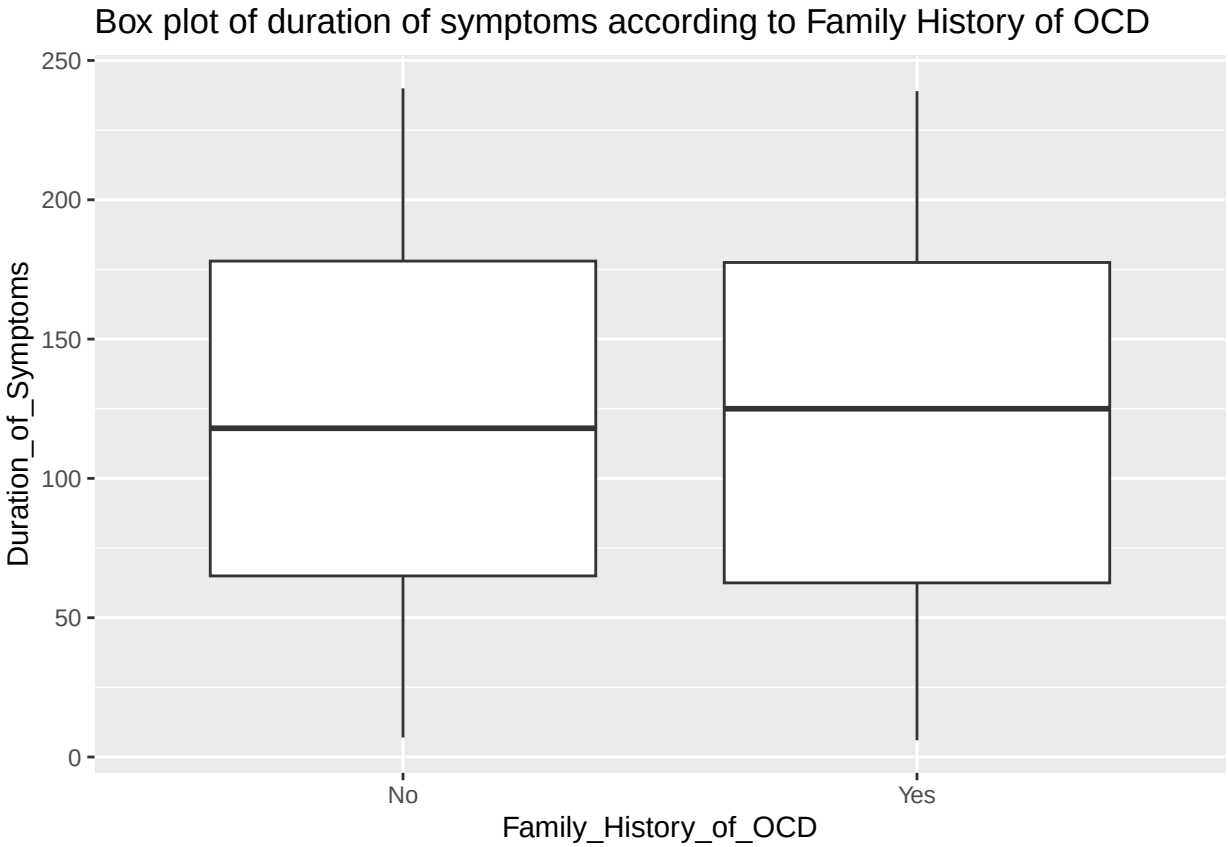
```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Previous_Diagnoses, fill=Previous_Diagnoses)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Previous_Diagnoses ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Previous Diagnoses")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



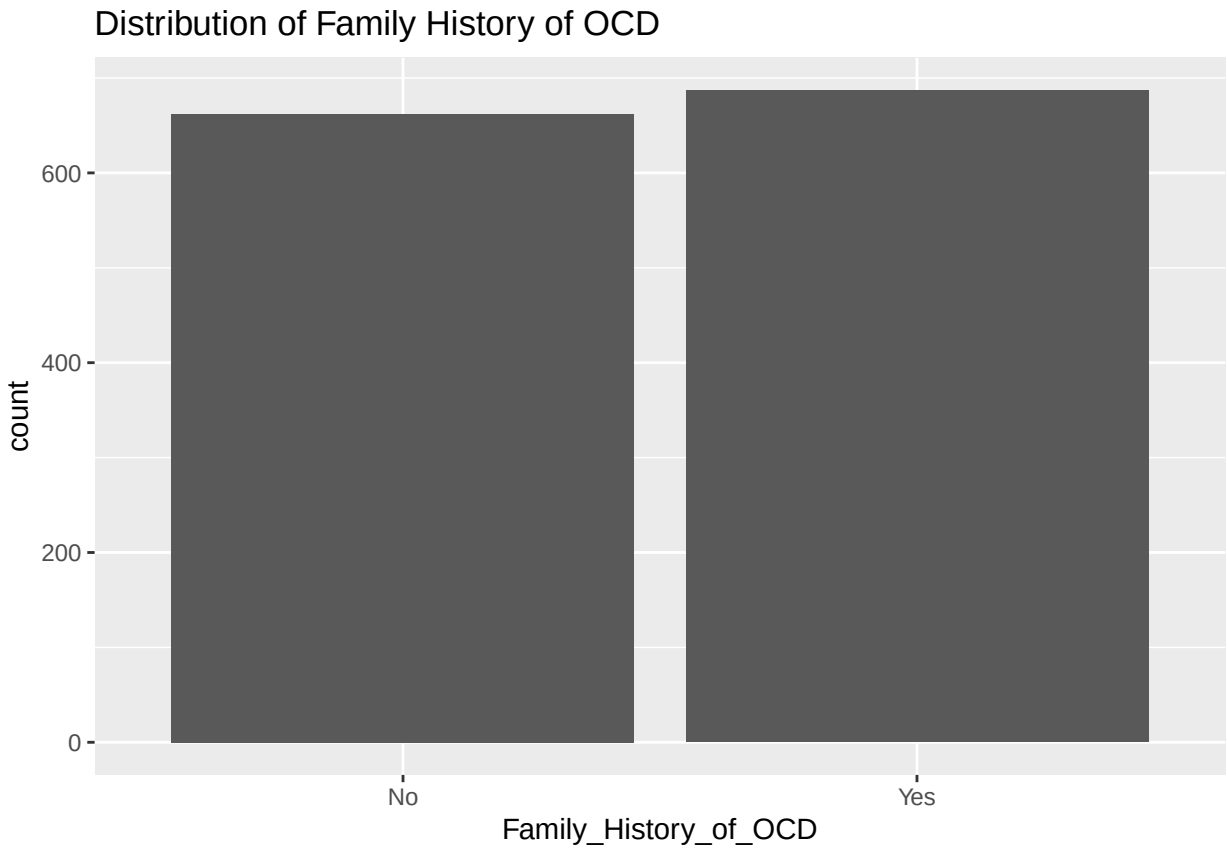
2.1.7 Family History of OCD

```
work_set %>% ggplot(aes(x=`Family_History_of_OCD`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Family History of OCD")
```



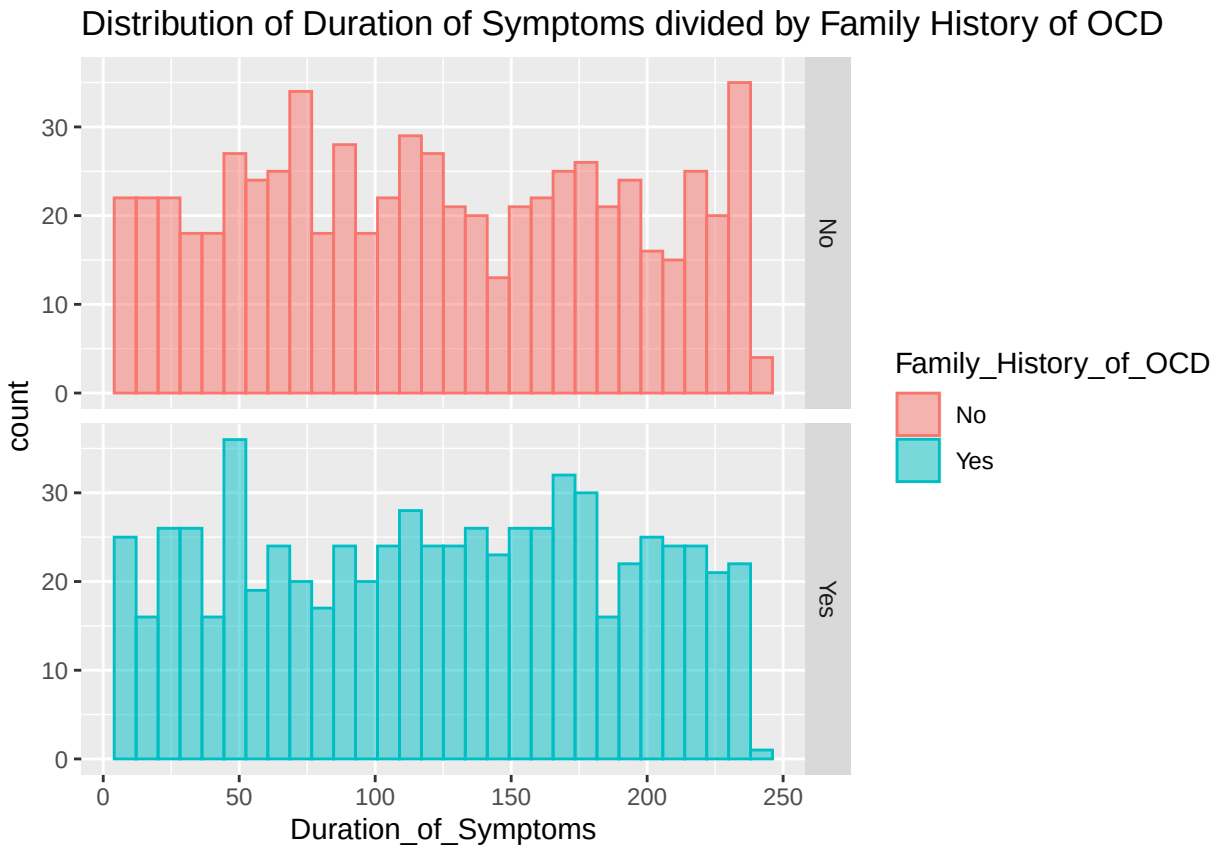
The average is slightly higher when there i a family history of OCD.

```
ggplot(work_set, aes(x=`Family_History_of_OCD`)) + geom_bar() + ggtitle("Distribution of Family History
```



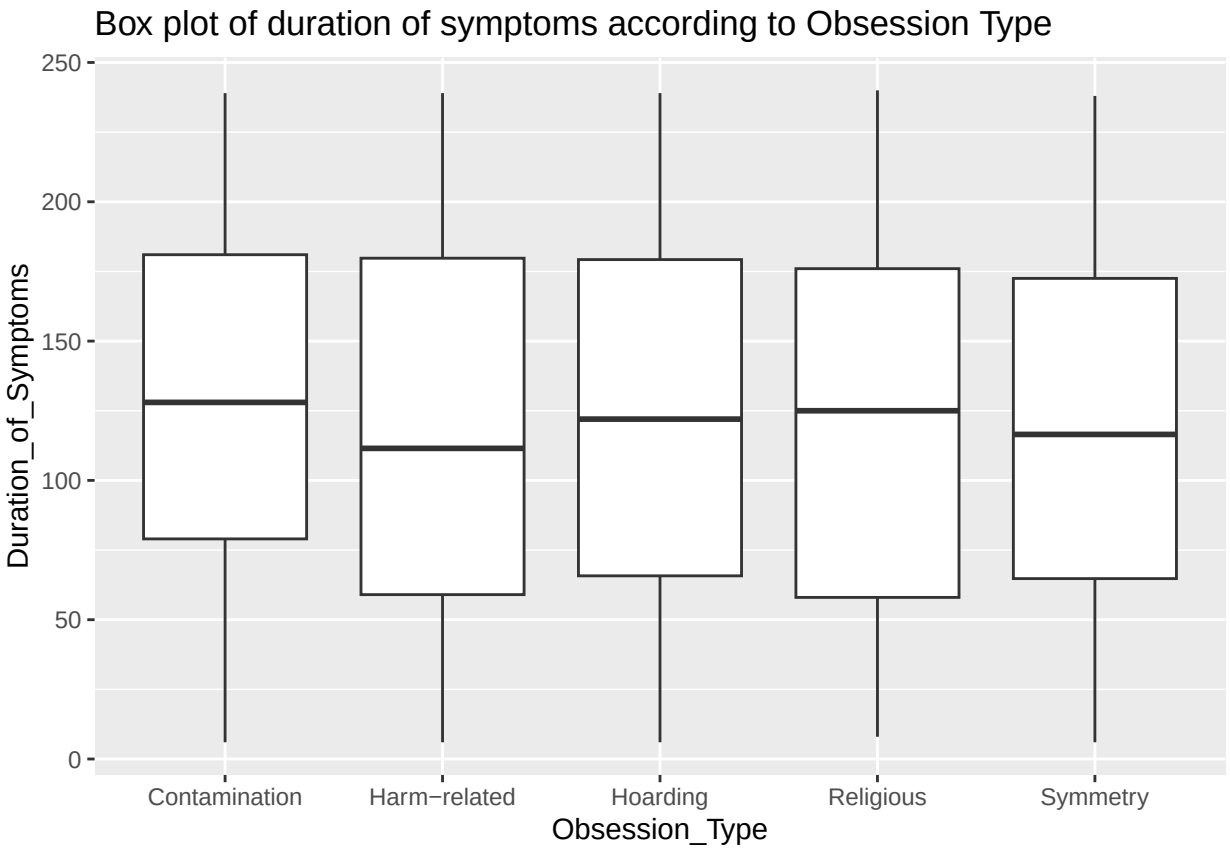

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Family_History_of_OCD, fill=Family_History_of_OCD)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Family_History_of_OCD ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Family History of OCD")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



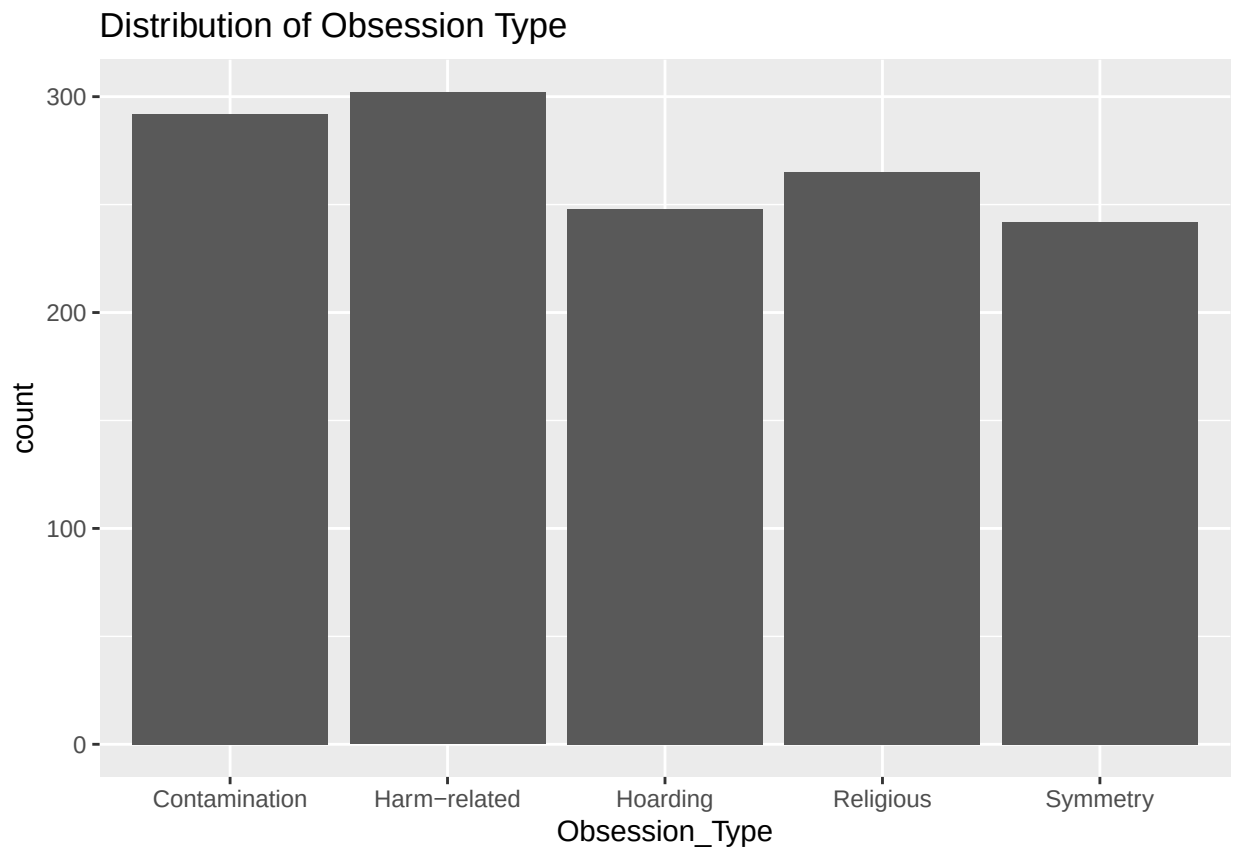
2.1.8 Obsession Type

```
work_set %>% ggplot(aes(x=`Obsession_Type`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Obsession Type")
```



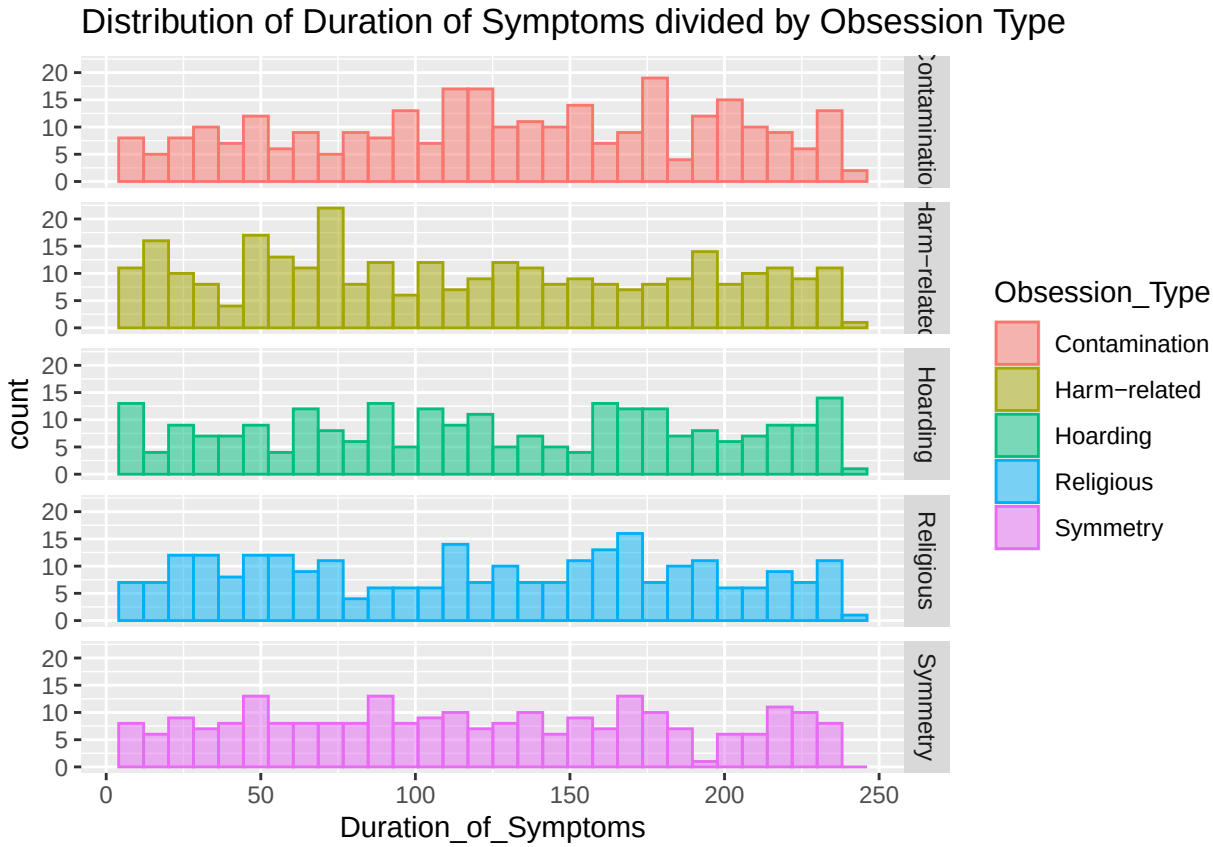
Some obsession types have lower average durations of symptoms, like Harm-related.

```
ggplot(work_set, aes(x=`Obsession_Type`)) + geom_bar() + ggtitle("Distribution of Obsession Type")
```



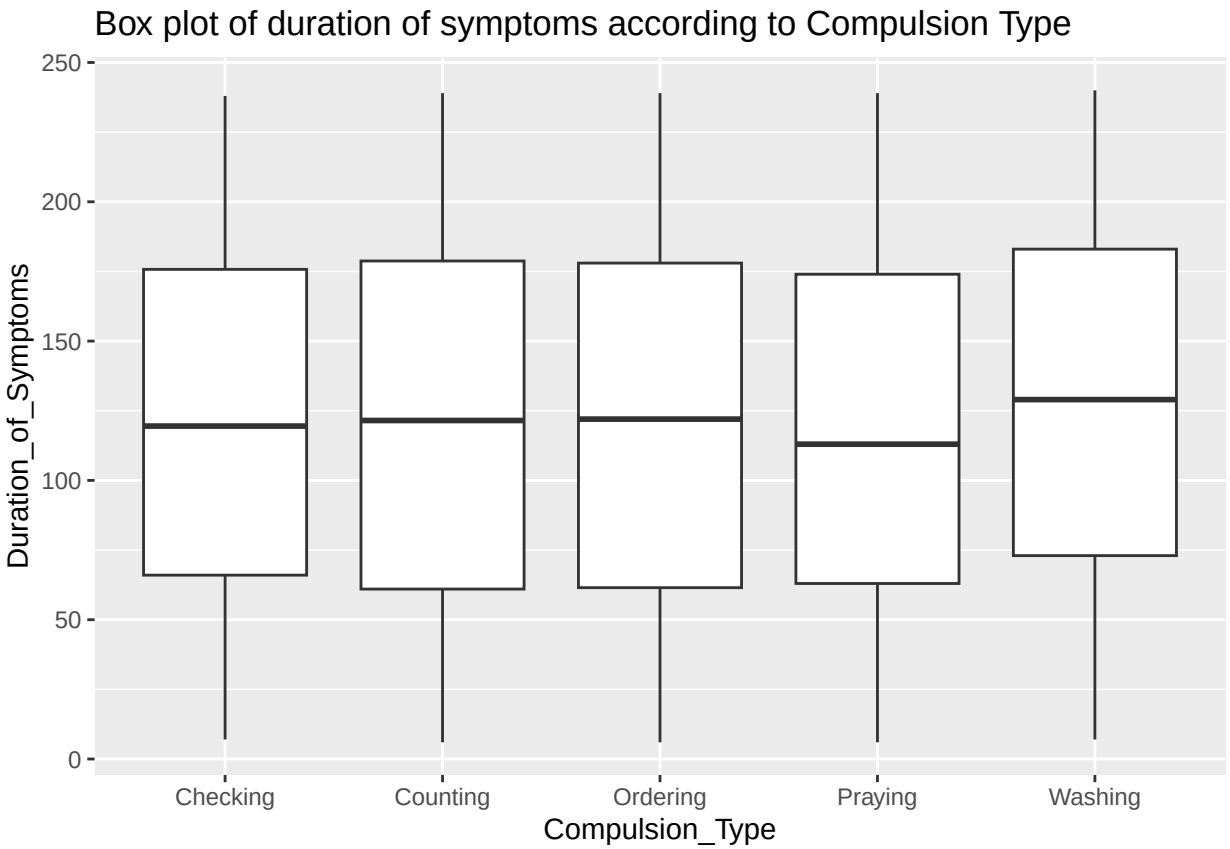

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Obsession_Type, fill=Obsession_Type)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Obsession_Type ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Obsession Type")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

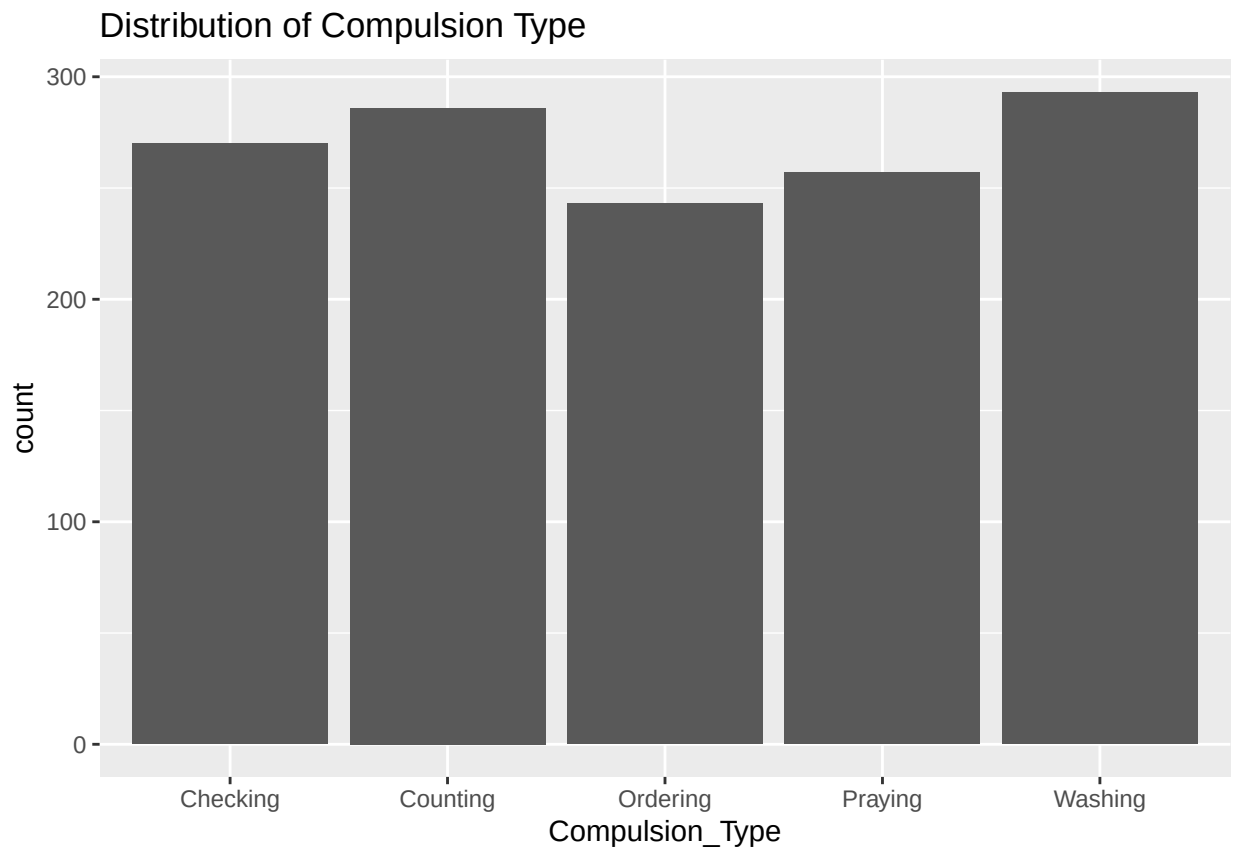


Compulsion Type

```
work_set %>% ggplot(aes(x=`Compulsion_Type`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Compulsion Type")
```

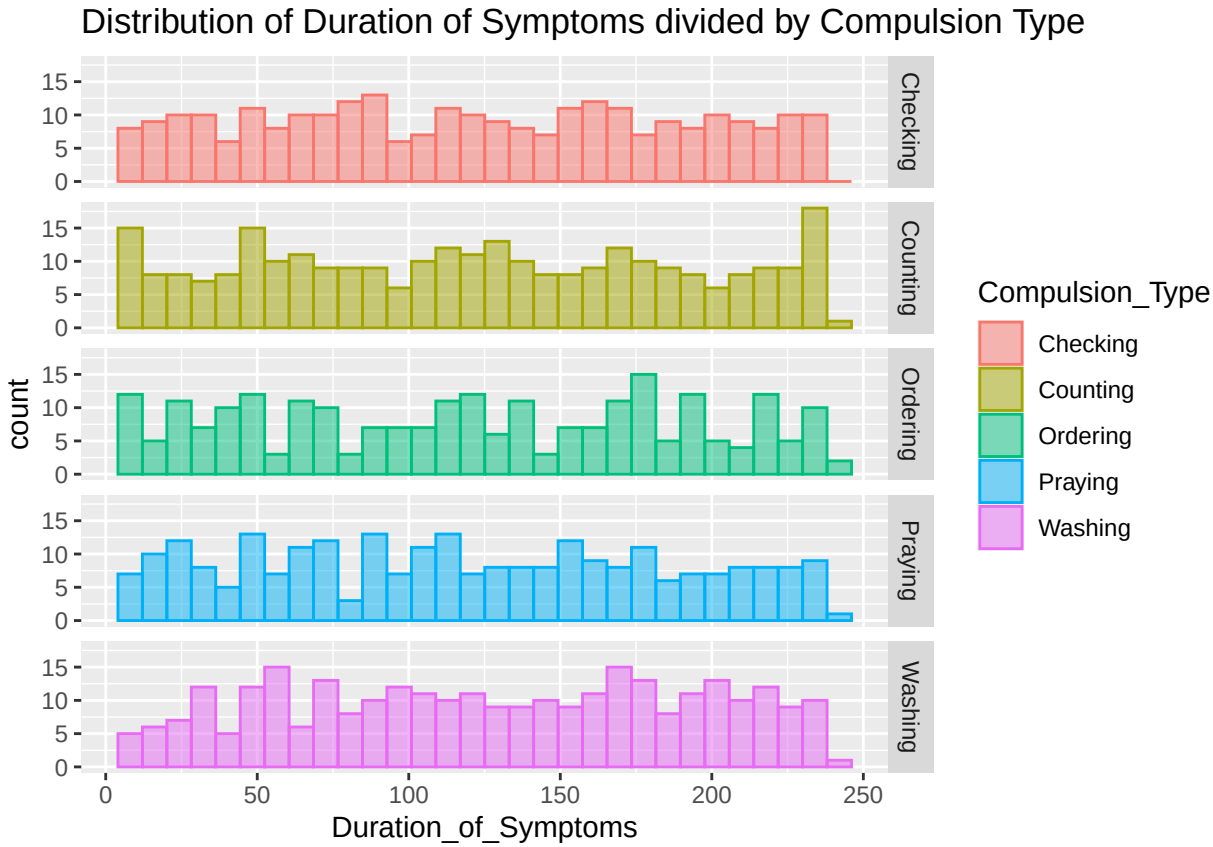



```
ggplot(work_set, aes(x=`Compulsion_Type`)) +
  geom_bar() +
  ggtitle("Distribution of Compulsion Type")
```



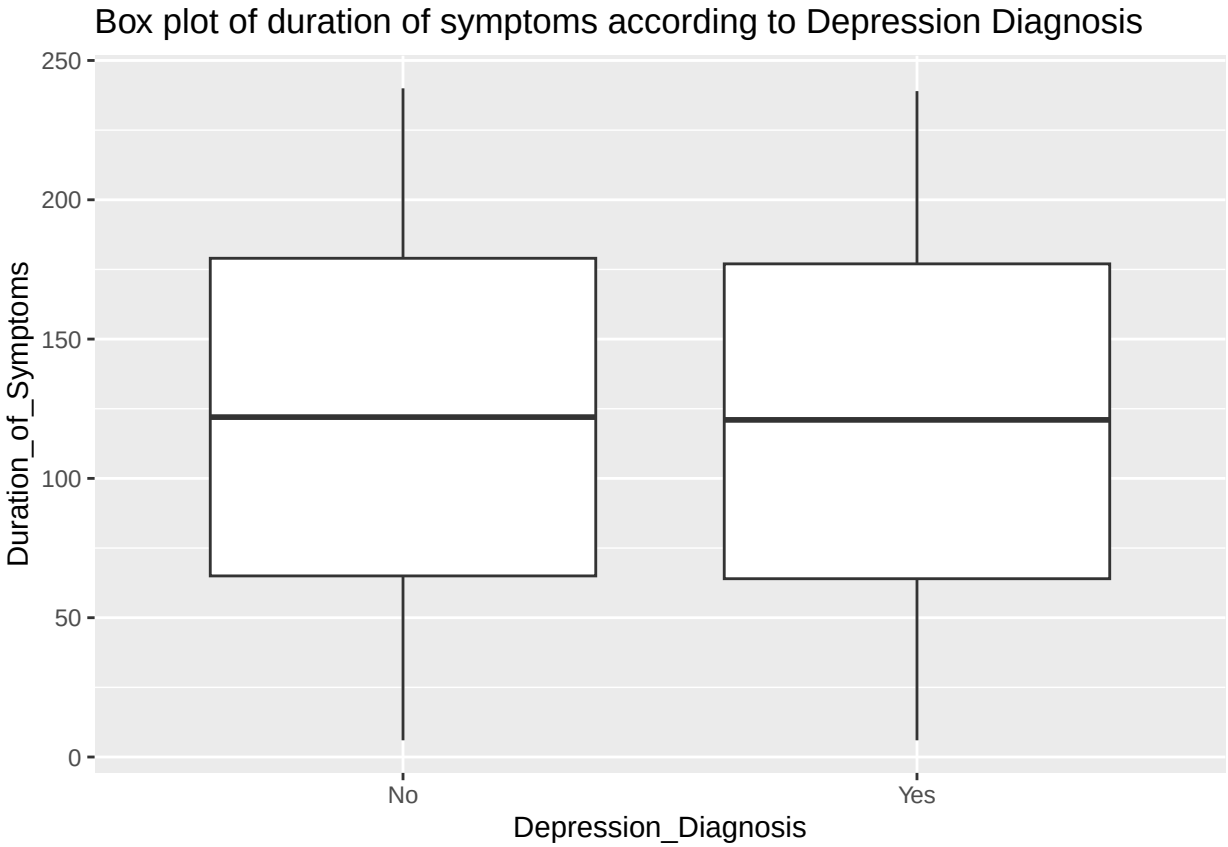

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Compulsion_Type, fill=Compulsion_Type)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Compulsion_Type ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Compulsion Type")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



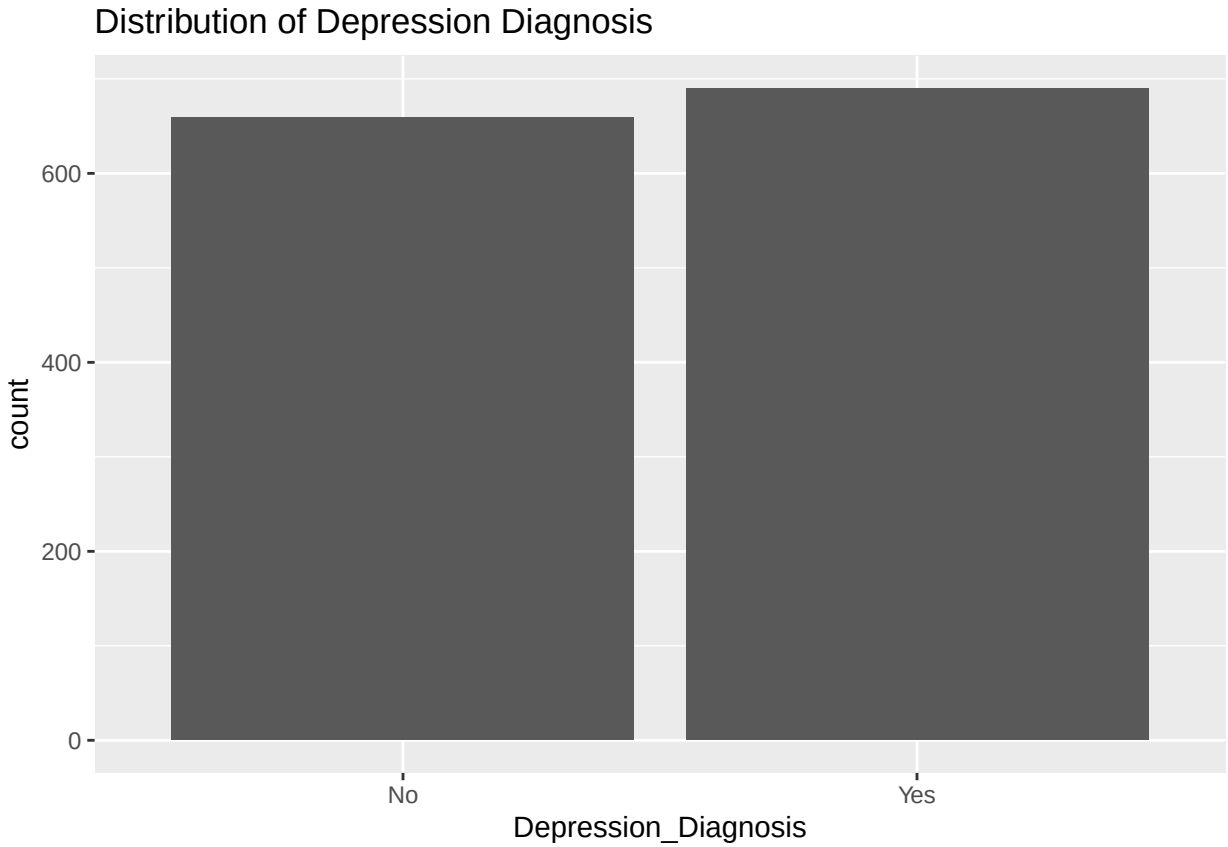
2.1.9 Depression Diagnosis

```
work_set %>% ggplot(aes(x=`Depression_Diagnosis`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Depression Diagnosis")
```



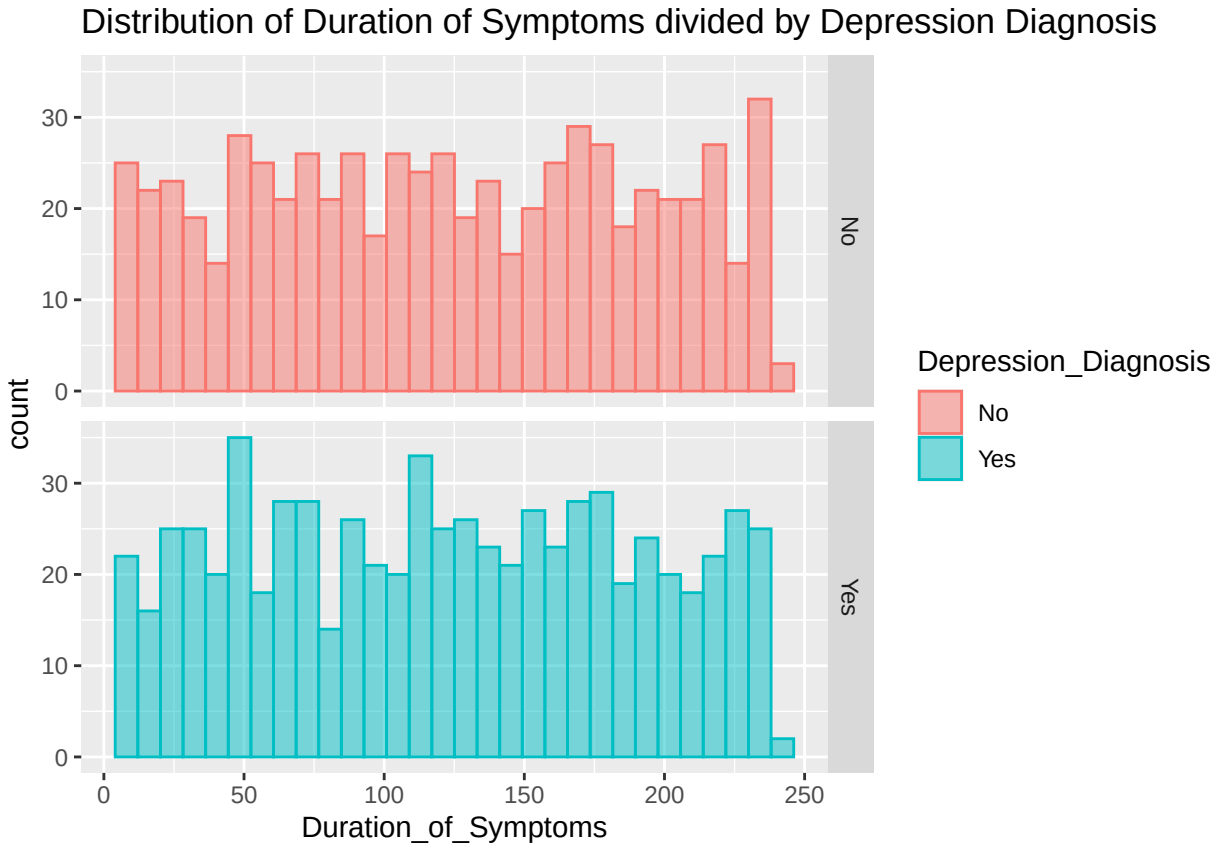
There is very little difference regarding the duration of symptoms between patients with a depression diagnosis and those who don't.

```
ggplot(work_set, aes(x=`Depression_Diagnosis`)) + geom_bar() + ggtitle("Distribution of Depression Diagn
```



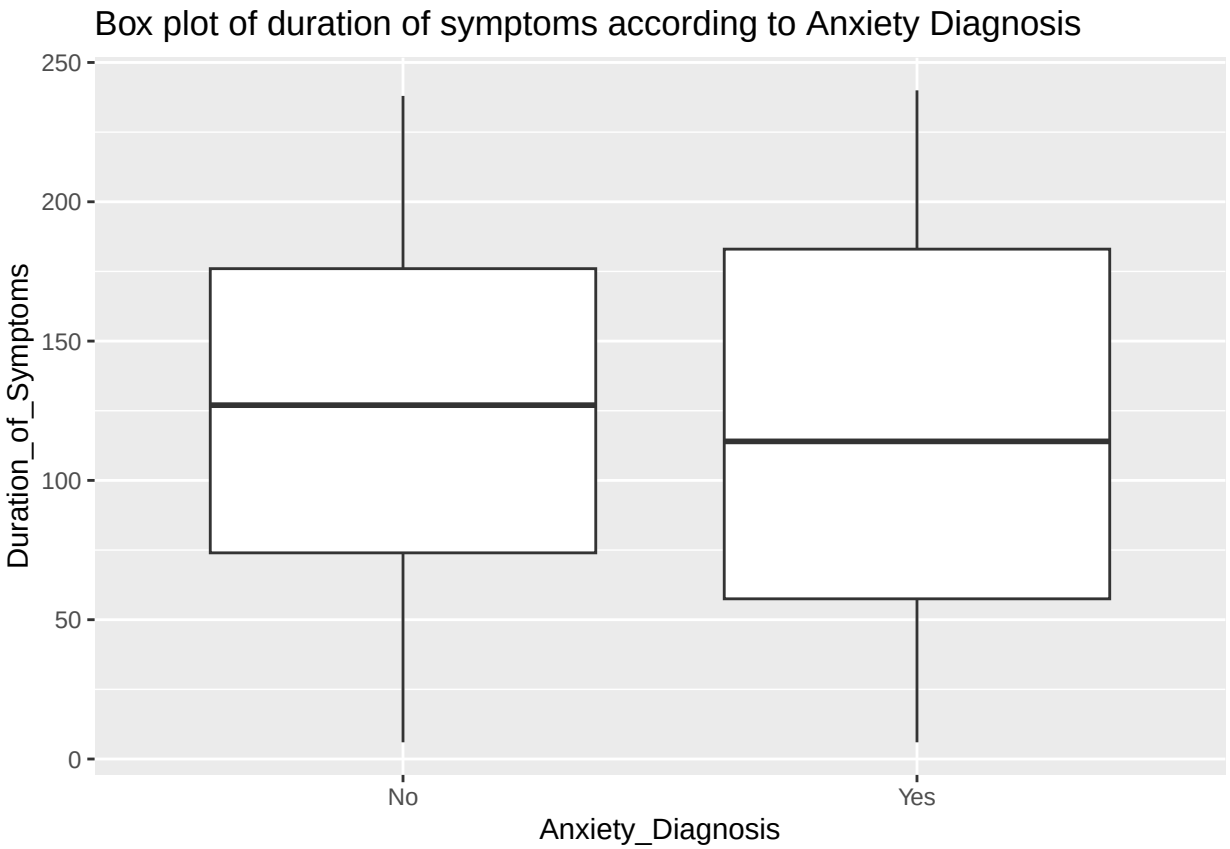

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Depression_Diagnosis, fill=Depression_Diagnosis)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Depression_Diagnosis ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Depression Diagnosis")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



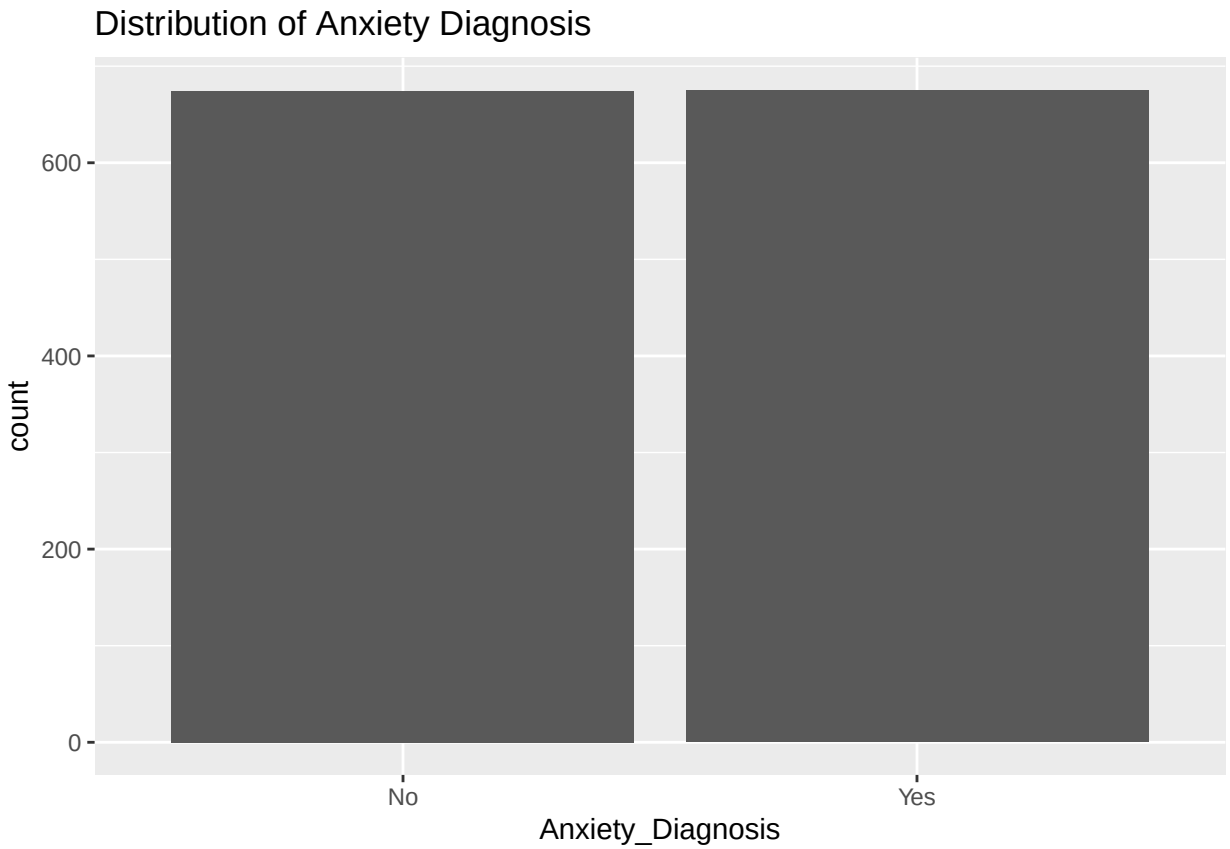
2.1.10 Anxiety Diagnosis

```
work_set %>% ggplot(aes(x=`Anxiety_Diagnosis`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Anxiety Diagnosis")
```



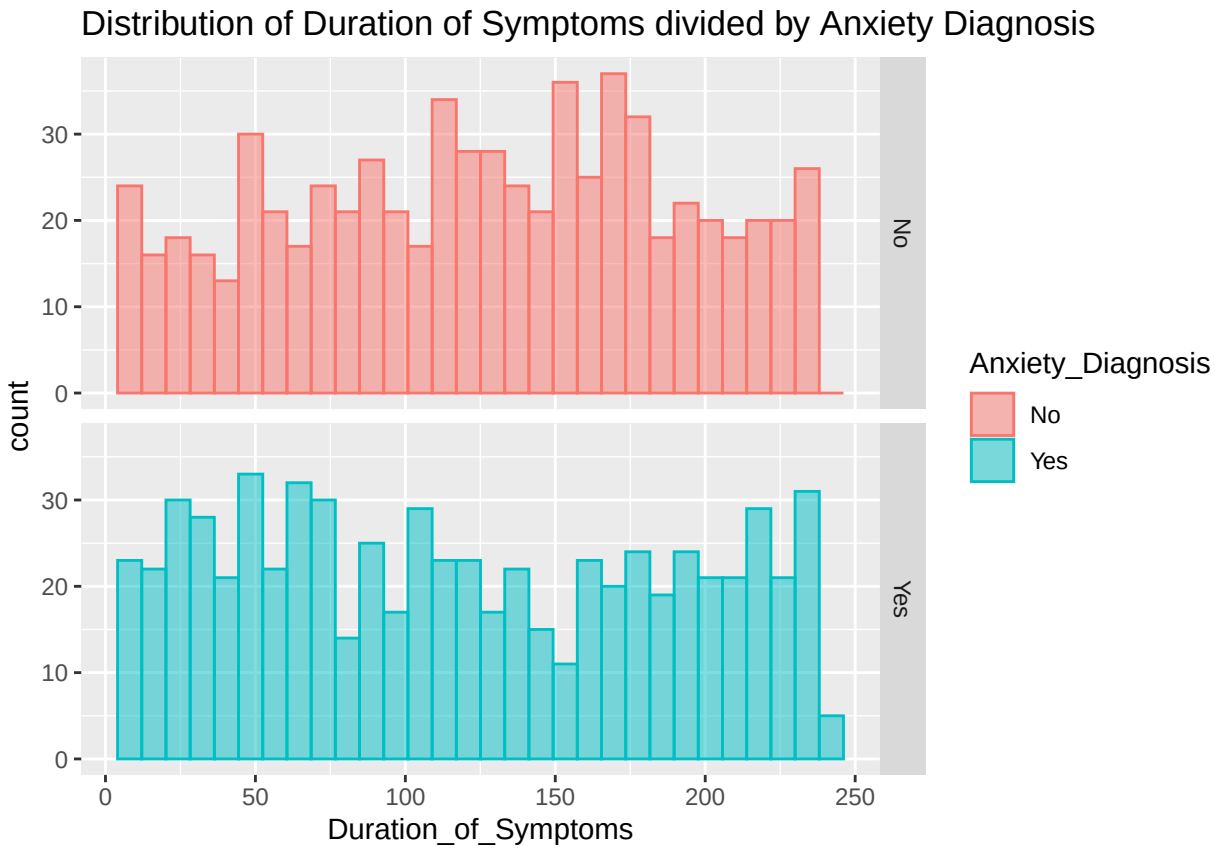
Patients with an Anxiety diagnosis seem to have a lower average but greater variance in their duration of symptoms.

```
ggplot(work_set, aes(x=`Anxiety_Diagnosis`)) + geom_bar() + ggtitle("Distribution of Anxiety Diagnosis"
```



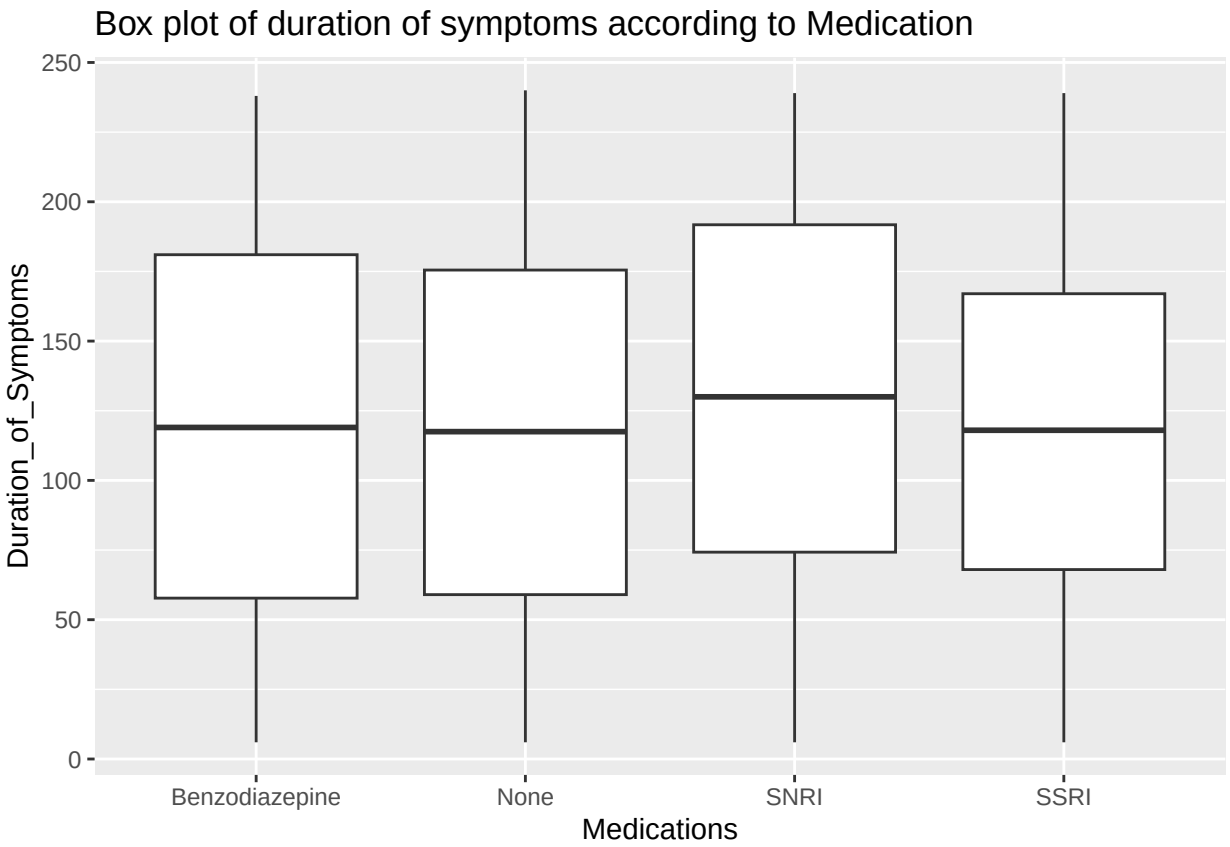

```
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Anxiety_Diagnosis, fill=Anxiety_Diagnosis)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Anxiety_Diagnosis ~ .) +
  ggtitle("Distribution of Duration of Symptoms divided by Anxiety Diagnosis")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



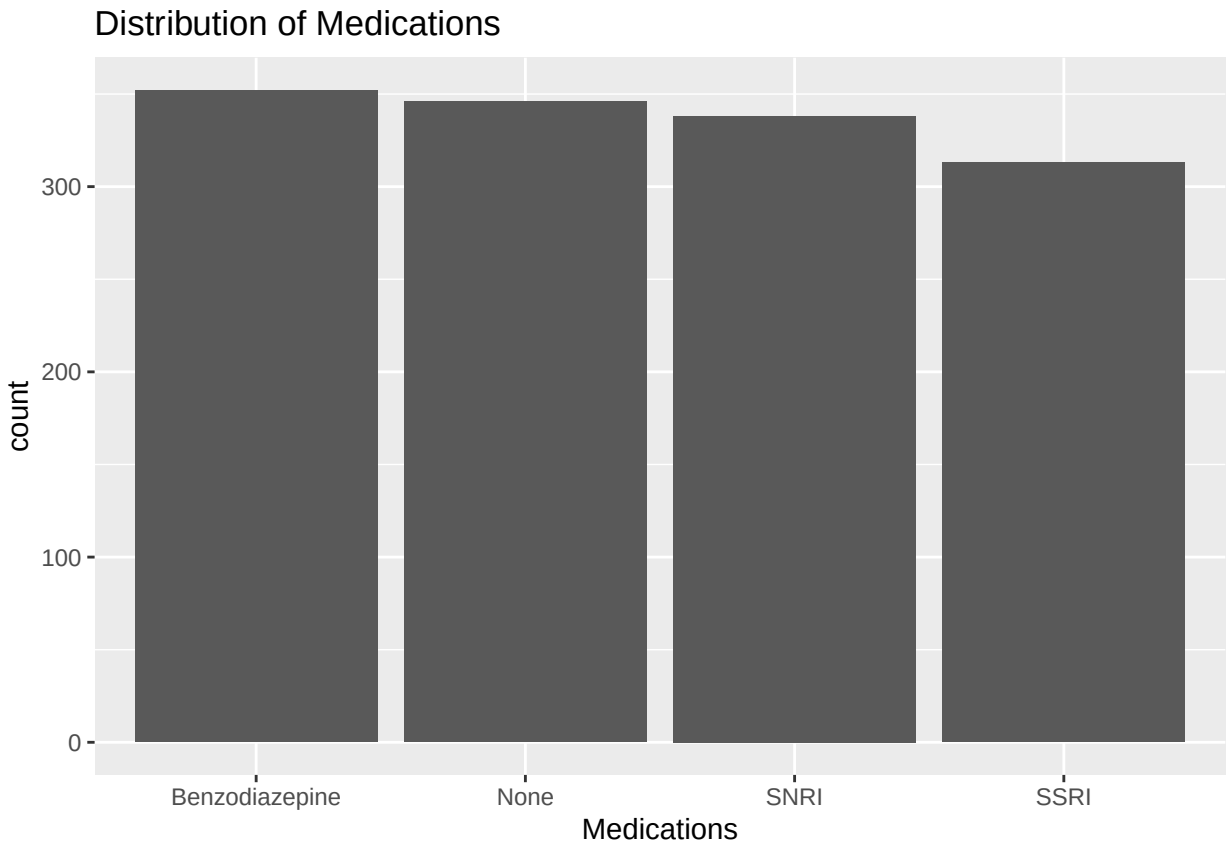
2.1.11 Medications

```
work_set %>% ggplot(aes(x=`Medications`, y=`Duration_of_Symptoms`)) +
  geom_boxplot()  +
  ggtitle("Box plot of duration of symptoms according to Medication")
```



Patients medicated with SNRI seem to have longer symptom duration than others.

```
ggplot(work_set, aes(x=`Medications`)) + geom_bar() + ggtitle("Distribution of Medications")
```



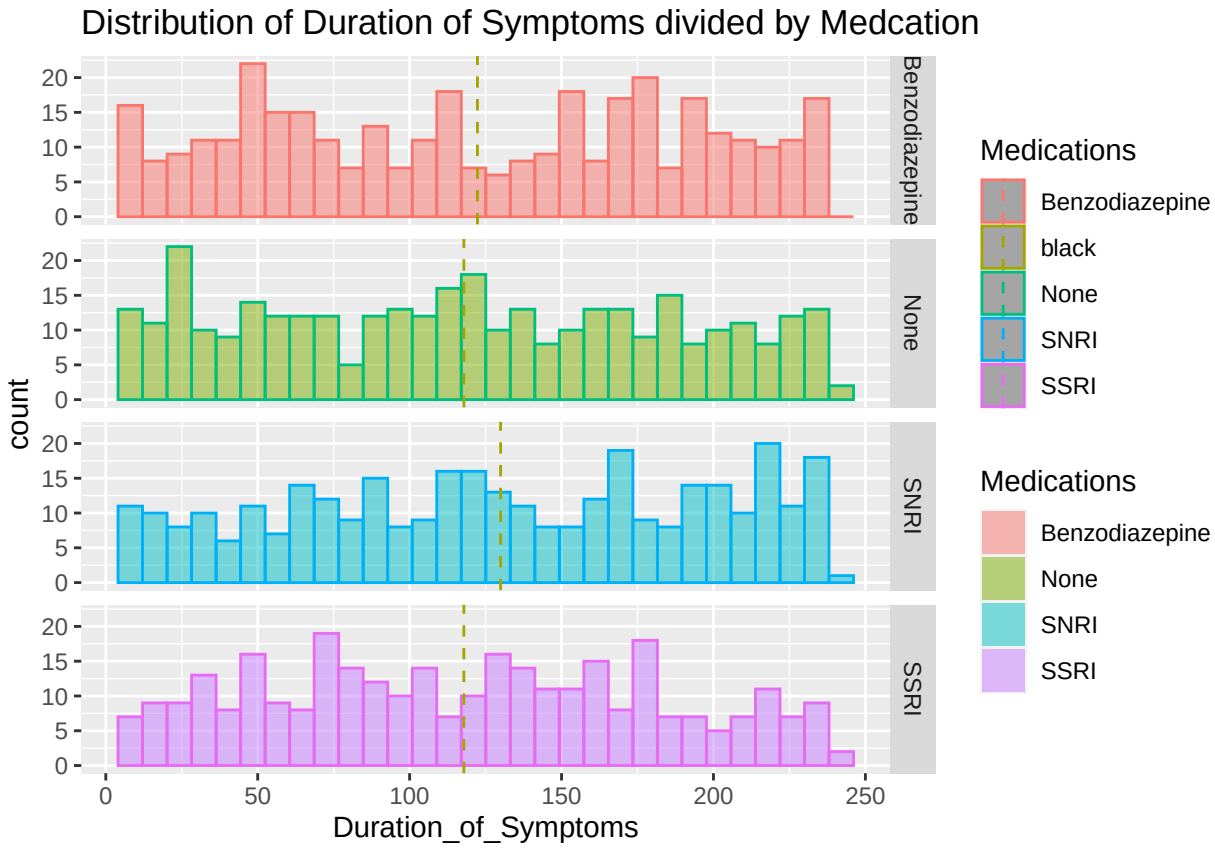

```
library(plyr)
```

```
## ------------------------------------------------------------------------------
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
## ------------------------------------------------------------------------------
##
## Attachement du package : 'plyr'
## L'objet suivant est masqué depuis 'package:here':
##
##     here
## Les objets suivants sont masqués depuis 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
## L'objet suivant est masqué depuis 'package:purrr':
##
##     compact
```

```
mu <- ddply(work_set, "Medications", summarise, grp.mean=mean(Duration_of_Symptoms))
ggplot(work_set, aes(x=Duration_of_Symptoms, color=Medications, fill=Medications)) +
  geom_histogram(alpha=0.5)+
  facet_grid(Medications ~ .)+
```

```
  geom_vline(
    data=mu,
    aes(
      xintercept=grp.mean,
      color="black"
    ),
    linetype="dashed"
    ) +
  ggtitle("Distribution of Duration of Symptoms divided by Medcation")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2.1.12 General Comment

After observing these graphs, we can notice that none of the variables shows a very significant impact over the duration of symptoms. The most interesting effects can be expected from combinations of factors or from the cummulative effect of multiple variables.

2.2 Data preparation

First we will encore our categorical variables. Gender, Ethnic, Marital Status, Previous Diagnoses, Family History of OCD, Obsession Type, Compulsion Type, Depression Diagnosis, Anxiety Diagnosis and Medications are all nominal. We will use one hot encoding. Education Level is an ordinal variable. We also eliminate the spaces in variable values to avoid future issues.

```
encoded_work_set <- work_set
#Binaries
```

```r
encoded_work_set <- mutate(encoded_work_set, Gender = if_else(Gender=="Female", 1, 0))
encoded_work_set <- mutate(encoded_work_set, Family_History_of_OCD  = if_else(Family_History_of_OCD=="N
encoded_work_set <- mutate(encoded_work_set, Depression_Diagnosis  = if_else(Depression_Diagnosis=="No"
encoded_work_set <- mutate(encoded_work_set, Anxiety_Diagnosis  = if_else(Anxiety_Diagnosis=="No",0,1))
#Nominals
Ethnicities <- c("African","Asian","Caucasian","Hispanic")
Education_Level <- c("High School","Some College","College Degree","Graduate Degree")
Marital_Statuses <- c("Divorced","Married","Single")
Previous_Diagnoses  <- c("GAD","MDD","Panic Disorder","PTSD","None")
Obsession_Types <- c("Contamination","Harm-related","Hoarding","Religious","Symmetry")
Compulsion_Types <- c("Checking", "Counting", "Ordering", "Praying", "Washing")
Medications <- c("Benzodiazepine","None","SNRI","SSRI")

for(index in 1:length(Ethnicities)){
    value <- Ethnicities[index]
    name = paste("Ethnicity", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Ethnicity,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}
for(index in 1:length(Marital_Statuses)){
    value <- Marital_Statuses[index]
    name = paste("Marital_Status", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Marital_Status,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}
for(index in 1:length(Previous_Diagnoses)){
    value <- Previous_Diagnoses[index]
    name = paste("Previous_Diagnosis", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Previous_Diagnoses,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}
for(index in 1:length(Obsession_Types)){
```

```
    value <- Obsession_Types[index]
    name = paste("Obsession_Type", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Obsession_Type,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}
for(index in 1:length(Compulsion_Types)){
    value <- Compulsion_Types[index]
    name = paste("Compulsion_Type", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Compulsion_Type,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}
for(index in 1:length(Medications)){
    value <- Medications[index]
    name = paste("Medications", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Medications,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}
for(index in 1:length(Education_Level)){
    value <- Education_Level[index]
    name = paste("Education_Level", value, sep = "_")
    encoded_work_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_work_set$Education_Level,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_work_set[name] <- detection
}

names(encoded_work_set)[names(encoded_work_set) == "Previous_Diagnosis_Panic Disorder"] <- "Previous_Di
```

```
names(encoded_work_set)[names(encoded_work_set) == "Obsession_Type_Harm-related"] <- "Obsession_Type_Ha
names(encoded_work_set)[names(encoded_work_set) == "Education_Level_High School"] <- "Education_Level_Hi
names(encoded_work_set)[names(encoded_work_set) == "Education_Level_Some College"] <- "Education_Level_S
names(encoded_work_set)[names(encoded_work_set) == "Education_Level_College Degree"] <- "Education_Level
names(encoded_work_set)[names(encoded_work_set) == "Education_Level_Graduate Degree"] <- "Education_Leve
```

We remove the old columns, since they have been encoded under new forms.

```
encoded_work_set <- encoded_work_set[,c(-1, -4,-5,-6,-9,-11,-12,-17)]
```

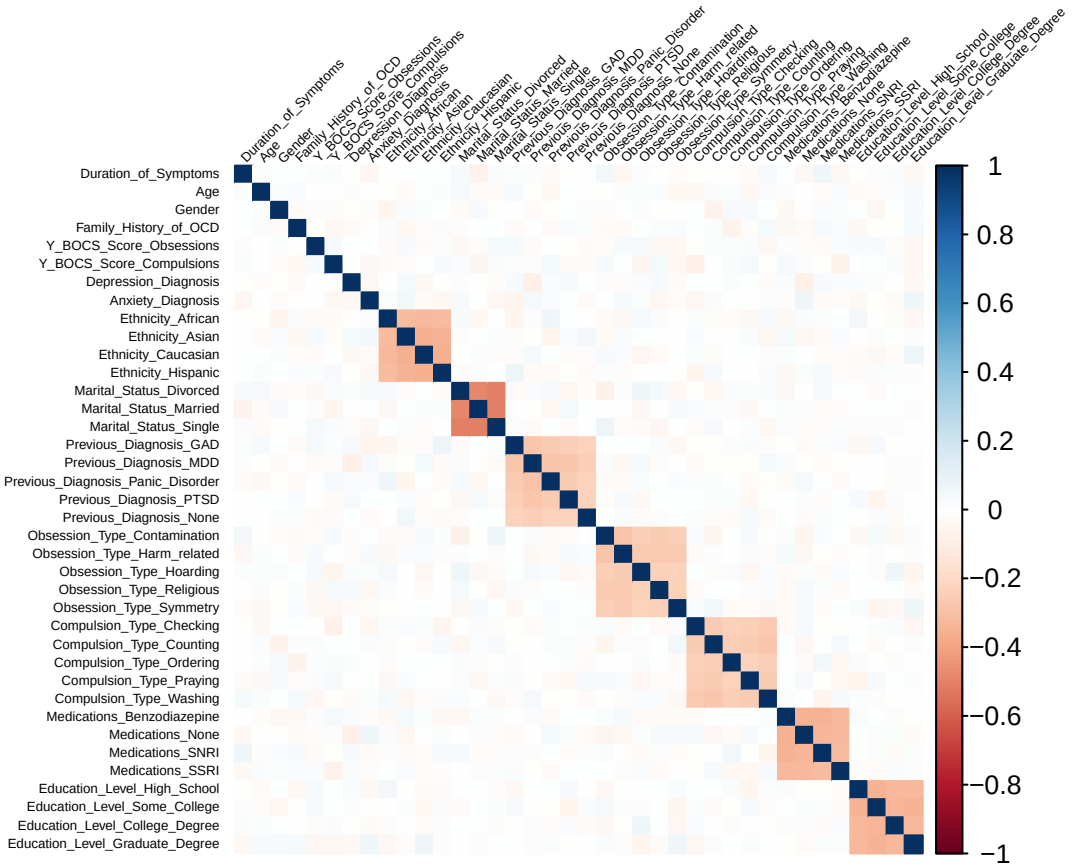
We also need to scale the data.

```
#create the scale
scale <- preProcess(encoded_work_set[,c(-4)], method=c("range"))
#apply it
scaled_work_set <- predict(scale, encoded_work_set)
```

2.3 Statistical analysis

After we transformed the dataset, we can apply some analysis that was not available. In particular the correlation plot.

```
subset_data <- scaled_work_set[, c("Duration_of_Symptoms","Age","Gender","Family_History_of_OCD","Y_BOCS
corrplot(cor(subset_data), method = "color", tl.col="black", tl.srt=45,tl.cex = 0.4)
```



We can see that the only major correlations between variables are in the categorical variables that we turned into binaries. Since they are mutually exclusive it is normal for them to be highly uncorrelated. However it seems that the duration of symptoms doesn't have much correlation with the rest. This suggests that the

available data is not very informative. This does not bode well for our goal, and we probably will not end with very precise predictions.

```
corrplot(cor(subset_data)[1,, drop=FALSE], method = "color", cl.pos='n', tl.col="black", tl.srt=45,tl.c
```

If we look specifically at the first line, we can see the correlations of our variables with the on thing we try to predict: the Duration of symptoms.

3 Methodology

We separate our work set between a training set and a testing set. Since we already have few data points (1349), we will be going for a small test set (5%).

```
train_test_index <- createDataPartition(y = scaled_work_set$Duration_of_Symptoms, times = 1, p = 0.05, l
dat_train <- scaled_work_set[-train_test_index,]
dat_test <- scaled_work_set[train_test_index,]
prediction_results <- dat_test[,4]
dat_test<- dat_test[,-4]
```

3.1 Models Exploration

Now we will try to implement several models in order to compare them. The end goal is to chose the one with the lowest RMSE.

3.1.1 Naive prediction

We start by taking the average duration of symptoms and using it as a generic prediction.

```r
avg <- mean(dat_train$Duration_of_Symptoms)
postResample(pred  = avg, obs  = prediction_results)
```

```
##     RMSE Rsquared      MAE
## 68.65346       NA 59.97167
```

```r
rmse_0 <- postResample(pred  = avg, obs  = prediction_results)[1]
```

We have a baseline of 68 month error with our RMSE. This is a lot, about 25% of our range of values.

3.1.2 Linear Model

Now, we will try a linear model with all our variables in order to predict the duration of symptoms with a bit more precision.

```r
linear_regression_model_1 <- lm(
  Duration_of_Symptoms ~ .,
  data = dat_train)
```

```r
linear_regression_prediction_1 <- predict(
  linear_regression_model_1,
  newdata = dat_test
)
```

```r
postResample(pred  = linear_regression_prediction_1, obs  = prediction_results)
```

```
##         RMSE     Rsquared          MAE
## 68.568238328  0.007391956 59.448579198
```

```r
rmse_1 <- postResample(pred  = linear_regression_prediction_1, obs  = prediction_results)[1]
```

Using a linear model, we get an RMSE of 68.568238328. This is barely an improvement over the 68.65346 of our naive method. We also notice that our Rsquared is very low. This is not a good prediction and we need to improve it.

3.1.3 Linear Model with interaction terms

We will try to add interaction terms to see if there is an improvement. First, we try to implement every possible interaction term. While this is not very good for prediction, it should be informative regarding the interactions that have the most effect.

```r
linear_regression_model_2 <- lm(
  Duration_of_Symptoms ~ .^2,
  data = dat_train)
```

```r
linear_regression_prediction_2 <- predict(
  linear_regression_model_2,
  newdata = dat_test
)
```

```
## Warning in predict.lm(linear_regression_model_2, newdata = dat_test):
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
postResample(pred  = linear_regression_prediction_2, obs  = prediction_results)
```

```
##       RMSE   Rsquared        MAE
## 74.5719417  0.0560419 59.8034184
```

```
rmse_2 <- postResample(pred  = linear_regression_prediction_2, obs  = prediction_results)[1]
```

We degraded our performances, but this is not unexpected. We added too many interaction variables (468 of them to be exact). The interesting part is that this will allow us to get the importance of he interaction terms and use the most interesting ones.

3.1.4 Linear Model with top terms

Let's isolate the top 20 interaction terms by importance and use them as terms for our linear model.

```
importances <- varImp(linear_regression_model_2, conditional=TRUE)
ordered_importances <-importances[order(importances[,"Overall"]), , drop = FALSE]
top_20 <- row.names(ordered_importances)[1:20]
```

```
formula <- ""
for(index in 1:20){
  formula <- paste(formula,top_20[index],sep = " + ")
}
formula  <- paste("Duration_of_Symptoms",formula,sep = " ~ ")
```

```
linear_regression_model_3 <- lm(
  formula,
  data = dat_train)
```

```
linear_regression_prediction_3 <- predict(
  linear_regression_model_3,
  newdata = dat_test
)
```

```
postResample(pred  = linear_regression_prediction_3, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
## 67.57878137  0.03388697 58.54744297
```

We see an improvement, at 67.57878137. This is still insufficient, and the Rsquared is still very low.

Lets try several numbers of "top n" interaction terms.

```
sequence <-seq( 5, 100,  5)
for (index_sequence in 1:length(sequence)){
  n <- sequence[index_sequence]
  top_n <- row.names(ordered_importances)[1:n]
  formula <- ""
  for(index_formula in 1:n){
    formula <- paste(formula,top_n[index_formula],sep = " + ")
  }
  formula  <- paste("Duration_of_Symptoms",formula,sep = " ~ ")
  linear_regression_model_3_n <- lm(
    formula,
    data = dat_train
    )
  linear_regression_prediction_3_n <- predict(
    linear_regression_model_3_n,
    newdata = dat_test
    )
  print(n)
  print(postResample(pred  = linear_regression_prediction_3_n, obs  = prediction_results)
```

```
)
}
```

```
## [1] 5
##        RMSE    Rsquared         MAE
## 67.99928885  0.05342812 59.13762336
## [1] 10
##        RMSE    Rsquared         MAE
## 67.85293114  0.03591865 58.79544661
## [1] 15
##        RMSE    Rsquared         MAE
## 67.59697796  0.04198904 58.43393658
## [1] 20
##        RMSE    Rsquared         MAE
## 67.57878137  0.03388697 58.54744297
## [1] 25
##        RMSE    Rsquared         MAE
## 67.55624462  0.03341588 58.16506185
## [1] 30
##        RMSE    Rsquared         MAE
## 67.31247173  0.04304108 58.02390455
## [1] 35
##        RMSE    Rsquared         MAE
## 67.87940506  0.02265715 58.61482341
## [1] 40
##        RMSE    Rsquared         MAE
## 67.92107998  0.02215616 58.82836401
## [1] 45
##        RMSE    Rsquared         MAE
## 67.90107725  0.02242932 58.65028977
## [1] 50
##        RMSE    Rsquared         MAE
## 67.79257229  0.02548507 58.45779269
## [1] 55
##        RMSE    Rsquared         MAE
## 67.88376212  0.02224225 58.74525334
## [1] 60
##        RMSE    Rsquared         MAE
## 68.44498567  0.01098393 59.09430742
## [1] 65
##         RMSE     Rsquared          MAE
## 68.856084810  0.004868454 59.421182677
## [1] 70
##         RMSE     Rsquared          MAE
## 68.942705040  0.005325535 60.205666275
## [1] 75
##        RMSE    Rsquared         MAE
## 68.22960911  0.01696964 59.83972818
## [1] 80
##        RMSE    Rsquared         MAE
## 67.59288778  0.03168016 59.36070689
## [1] 85
##        RMSE    Rsquared         MAE
## 67.40819105  0.03649711 59.14829829
```

```
## [1] 90
##        RMSE    Rsquared         MAE
## 67.36545122  0.03865537 58.95746997
## [1] 95
##        RMSE    Rsquared         MAE
## 68.27502811  0.01959654 59.47933298
## [1] 100
##       RMSE   Rsquared        MAE
## 68.6049085  0.0146206 59.6905142
```

After some tweaking, we get a better result for the top 30 This is currently our best score, with an RMSE of 67.31247173 .

```
top_30 <- row.names(ordered_importances)[1:30]
formula <- ""
for(index in 1:30){
  formula <- paste(formula,top_30[index],sep = " + ")
}
formula  <- paste("Duration_of_Symptoms",formula,sep = " ~ ")
```

```
linear_regression_model_3 <- lm(
  formula,
  data = dat_train)
```

```
linear_regression_prediction_3 <- predict(
  linear_regression_model_3,
  newdata = dat_test
)
```

```
postResample(pred  = linear_regression_prediction_3, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
## 67.31247173  0.04304108 58.02390455
```

```
rmse_3  <- postResample(pred  = linear_regression_prediction_3, obs  = prediction_results)[1]
```

3.1.5 Linear Model with initial variable and top interactions

Now, we might be interested in using the top interaction terms but also using all of the original variables.

```
top_30 <- row.names(ordered_importances)[1:30]
formula_2 <- ".+"
for(index in 1:30){
  formula_2 <- paste(formula_2,top_30[index],sep = " + ")
}
formula_2  <- paste("Duration_of_Symptoms",formula_2,sep = " ~ ")
```

```
linear_regression_model_4 <- lm(
  formula_2,
  data = dat_train)
```

```
linear_regression_prediction_4 <- predict(
  linear_regression_model_4,
  newdata = dat_test
)
```

```
postResample(pred  = linear_regression_prediction_4, obs  = prediction_results)
```

```
##         RMSE     Rsquared          MAE
## 68.622035662  0.008425085 59.355632545
```

```
rmse_4 <- postResample(pred  = linear_regression_prediction_4, obs  = prediction_results)[1]
```

After considering the RMSE, this is not particularly interesting.

3.1.6 Multinomial regression

Now, we will compare our linear model to an alternative: a multinomial model. For this, we will use squared versions of the numeric variables and add them to the mix.

```
#Create multinomial terms
dat_train_binomial <- dat_train
dat_train_binomial$Age_squared <- dat_train$Age^2
dat_train_binomial$Y_BOCS_Score_Obsessions_squared <- dat_train_binomial$Y_BOCS_Score_Obsessions^2
dat_train_binomial$Y_BOCS_Score_Compulsions_squared <- dat_train_binomial$Y_BOCS_Score_Compulsions^2
dat_test_binomial <- dat_test
dat_test_binomial$Age_squared <- dat_test$Age^2
dat_test_binomial$Y_BOCS_Score_Obsessions_squared <- dat_test_binomial$Y_BOCS_Score_Obsessions^2
dat_test_binomial$Y_BOCS_Score_Compulsions_squared <- dat_test_binomial$Y_BOCS_Score_Compulsions^2
```

```
binomial_regression_model_1 <- lm(
  "Duration_of_Symptoms ~ .",
  data = dat_train_binomial)
```

```
binomial_regression_prediction_1 <- predict(
  binomial_regression_model_1,
  newdata = dat_test_binomial
)
```

```
postResample(pred  = binomial_regression_prediction_1, obs  = prediction_results)
```

```
##         RMSE     Rsquared          MAE
## 68.558315902  0.008458194 59.426512449
```

```
rmse_5 <- postResample(pred  = binomial_regression_prediction_1, obs  = prediction_results)[1]
```

We get a slightly worse result than our champion (the previous one was at 67.31247173 / 0.04304108 / 58.02390455). But maybe we can get better interaction terms.

3.1.7 Multinomial regression with interaction terms

```
binomial_regression_model_2 <- lm(
  "Duration_of_Symptoms ~ .^2",
  data = dat_train_binomial)
```

```
binomial_regression_prediction_2 <- predict(
  binomial_regression_model_2,
  newdata = dat_test_binomial
)
```

```
## Warning in predict.lm(binomial_regression_model_2, newdata =
## dat_test_binomial): prediction from rank-deficient fit; attr(*, "non-estim")
## has doubtful cases
```

```
postResample(pred  = binomial_regression_prediction_2, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
```

```
## 82.40139606  0.04484384 67.63683937
```

```
rmse_6 <- postResample(pred  = binomial_regression_prediction_2, obs  = prediction_results)[1]
```

The results are very bad. Let's isolate the top 30 interactions as we did previously.

3.1.8 Multinomial regression with top interaction terms

```
importances_binomial <- varImp(binomial_regression_model_2, conditional=TRUE)
#ordered_importances_binomial <-importances_binomial %>% arrange(., desc(Overall))
ordered_importances_binomial <-importances[order(importances[,"Overall"]), , drop = FALSE]

top_30_binomial <- row.names(ordered_importances_binomial)[1:30]
formula_binomial <- ""
for(index in 1:30){
  formula_binomial <- paste(formula_binomial,top_30_binomial[index],sep = " + ")
}
formula_binomial  <- paste("Duration_of_Symptoms",formula_binomial,sep = " ~ ")
```

```
binomial_regression_model_3 <- lm(
  formula_binomial,
  data = dat_train_binomial)
```

```
binomial_regression_prediction_3 <- predict(
  binomial_regression_model_3,
  newdata = dat_test_binomial
)
```

```
postResample(pred  = binomial_regression_prediction_3, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
## 67.31247173  0.04304108 58.02390455
```

We match the current result of 67.31247173 / 0.04304108 / 58.02390455. This suggests that none of the squared values ended up in our top 30 and benefited us. We inherited 30 from the linear model. Lets try other numbers

```
for (index_sequence in 1:length(sequence)){
  n <- sequence[index_sequence]
  top_n <- row.names(ordered_importances_binomial)[1:n]
  formula <- ""
  for(index_formula in 1:n){
    formula <- paste(formula,top_n[index_formula],sep = " + ")
  }
  formula  <- paste("Duration_of_Symptoms",formula,sep = " ~ ")
  binomial_regression_model_3_n <- lm(
    formula,
    data = dat_train_binomial
    )
  binomial_regression_prediction_3_n <- predict(
    binomial_regression_model_3_n,
    newdata = dat_test_binomial
    )
  print(n)
  print(postResample(pred  = binomial_regression_prediction_3_n, obs  = prediction_results)
)
```

```
}
```

```
## [1] 5
##        RMSE    Rsquared         MAE
## 67.99928885  0.05342812 59.13762336
## [1] 10
##        RMSE    Rsquared         MAE
## 67.85293114  0.03591865 58.79544661
## [1] 15
##        RMSE    Rsquared         MAE
## 67.59697796  0.04198904 58.43393658
## [1] 20
##        RMSE    Rsquared         MAE
## 67.57878137  0.03388697 58.54744297
## [1] 25
##        RMSE    Rsquared         MAE
## 67.55624462  0.03341588 58.16506185
## [1] 30
##        RMSE    Rsquared         MAE
## 67.31247173  0.04304108 58.02390455
## [1] 35
##        RMSE    Rsquared         MAE
## 67.87940506  0.02265715 58.61482341
## [1] 40
##        RMSE    Rsquared         MAE
## 67.92107998  0.02215616 58.82836401
## [1] 45
##        RMSE    Rsquared         MAE
## 67.90107725  0.02242932 58.65028977
## [1] 50
##        RMSE    Rsquared         MAE
## 67.79257229  0.02548507 58.45779269
## [1] 55
##        RMSE    Rsquared         MAE
## 67.88376212  0.02224225 58.74525334
## [1] 60
##        RMSE    Rsquared         MAE
## 68.44498567  0.01098393 59.09430742
## [1] 65
##         RMSE     Rsquared          MAE
## 68.856084810  0.004868454 59.421182677
## [1] 70
##         RMSE     Rsquared          MAE
## 68.942705040  0.005325535 60.205666275
## [1] 75
##        RMSE    Rsquared         MAE
## 68.22960911  0.01696964 59.83972818
## [1] 80
##        RMSE    Rsquared         MAE
## 67.59288778  0.03168016 59.36070689
## [1] 85
##        RMSE    Rsquared         MAE
## 67.40819105  0.03649711 59.14829829
## [1] 90
```

```
##        RMSE    Rsquared         MAE
## 67.36545122  0.03865537 58.95746997
## [1] 95
##        RMSE    Rsquared         MAE
## 68.27502811  0.01959654 59.47933298
## [1] 100
##       RMSE   Rsquared        MAE
## 68.6049085  0.0146206 59.6905142
```

We get our best performance at 30. There is no change.

```
top_30_binomial <- row.names(ordered_importances_binomial)[1:30]
formula_binomial <- ""
for(index in 1:30){
  formula_binomial <- paste(formula_binomial,top_30_binomial[index],sep = " + ")
}
formula_binomial  <- paste("Duration_of_Symptoms",formula_binomial,sep = " ~ ")
```

```
binomial_regression_model_3 <- lm(
  formula_binomial,
  data = dat_train_binomial)
```

```
binomial_regression_prediction_3 <- predict(
  binomial_regression_model_3,
  newdata = dat_test_binomial
)
```

```
postResample(pred  = binomial_regression_prediction_3, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
## 67.31247173  0.04304108 58.02390455
```

```
rmse_7 <- postResample(pred  = binomial_regression_prediction_3, obs  = prediction_results)[1]
```

We can conclude that a binomial model does not bring anything particularly interesting over the linear model.

3.1.9 Support Vector Regression‘

Another model we might use is a support vector model geared toward regression.

```
svm_model_1 <- svm(Duration_of_Symptoms ~ . , dat_train)

svm_prediction_1 <- predict(svm_model_1, newdata = dat_test)
postResample(pred  = svm_prediction_1, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
## 66.58566545  0.07094881 55.25426508
```

```
rmse_8 <- postResample(pred  = svm_prediction_1, obs  = prediction_results)[1]
```

With an RMSE of 66.58566545, the support vector model performs better than both the linear and binomial models.

3.1.10 Optimized Support Vector Regression‘

We can improve our model by tuning some parameters

```
gamma_options = c(0.01,0.1, 1,5,10)
cost_options = c(0.01,0.1, 1,5,10)
```

```
best_RMSE <- 100
best_gamma <- 0
best_cost <- 0
for(index_gamma in 1:length(gamma_options)){
  for(index_cost in 1:length(cost_options)){
    gamma = gamma_options[index_gamma]
    cost = cost_options[index_cost]
    svm_model <-svm(
      Duration_of_Symptoms~.,
      data = dat_train,
      gamma = gamma,
      cost = cost
      )
    svm_prediction <- predict(svm_model, newdata = dat_test)
    RMSE <- postResample(pred  = svm_prediction, obs  = prediction_results)[1]
    if(RMSE < best_RMSE){
      best_RMSE <- RMSE
      best_gamma <- gamma
      best_cost <- cost
    }
  }
}
best_RMSE
```

```
##     RMSE
## 66.49303
```

```
best_gamma
```

```
## [1] 0.01
```

```
best_cost
```

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

```
svm_model_2 <-svm(
  Duration_of_Symptoms~.,
  data = dat_train,
  gamma = 0.01,
  cost = 1)
svm_prediction_2 <- predict(svm_model_2, newdata = dat_test)
postResample(pred  = svm_prediction_2, obs  = prediction_results)
```

```
##       RMSE   Rsquared        MAE
## 66.4930270  0.0626489 56.0075671
```

```
rmse_9 <- postResample(pred  = svm_prediction_2, obs  = prediction_results)[1]
```

We slightly improved on the previous result, from 66.58566545 to 66.4930270. This is a marginal improvement considering the scale of our RMSE.

3.1.11 Random forest regression

Another option is the Random forest. We implement it here, with a thousand trees.

```
random_forest_model_1 <- randomForest(Duration_of_Symptoms ~ ., data=dat_train, ntree=1000, importance="
```

```
random_forest_prediction_1 <- predict(random_forest_model_1, newdata = dat_test)
postResample(pred  = random_forest_prediction_1, obs  = prediction_results)
```

```
##        RMSE    Rsquared         MAE
## 67.34368838  0.03844803 57.81846356
```

```
rmse_10 <- postResample(pred  = random_forest_prediction_1, obs  = prediction_results)[1]
```

Random forest does not perform as well as our current champion, the SVM.

3.1.12 ANN regression

Te last model we experiment with will be the Artificial Neural Network. We use the h2o library. A series of test showed that due to our small number of data points we overtrain within a very small number of epochs. 3 hidden layers of 20 neurons also showed the best result.

```
#Initialize h2o
h2o.init(nthreads = -1)
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         31 minutes 3 seconds
##     H2O cluster timezone:       America/Toronto
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.42.0.2
##     H2O cluster version age:    4 months and 8 days
##     H2O cluster name:           H2O_started_from_R_Admin_coj423
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   7.78 GB
##     H2O cluster total cores:    16
##     H2O cluster allowed cores:  16
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     R Version:                  R version 4.3.1 (2023-06-16 ucrt)
```

```
## Warning in h2o.clusterInfo():
## Your H2O cluster version is (4 months and 8 days) old. There may be a newer version available.
## Please download and install the latest version from: https://h2o-release.s3.amazonaws.com/h2o/latest
```

```
#create model
ann_model_1 = h2o.deeplearning(
  y = 'Duration_of_Symptoms',
  training_frame = as.h2o(dat_train),
  activation = 'Rectifier',
  hidden = c(20, 20, 20),
  epochs = 5
  )
```

```
##   |                                                                              |
##   |                                                                              |
```

```
#predict values
ann_prediction_1 = h2o.predict(
  ann_model_1,
```

```
  newdata = as.h2o(dat_test)
  )
```

```
##   |                                                                      |
##   |                                                                      |
```

```
ann_prediction_1 = as.vector(ann_prediction_1)
#calculate RMSE
postResample(pred  = ann_prediction_1, obs  = prediction_results)
```

```
##         RMSE     Rsquared          MAE
## 7.235851e+01 5.471508e-04 6.149155e+01
```

```
rmse_11 <- postResample(pred  = ann_prediction_1, obs  = prediction_results)[1]
```

After some tweaking we reached an RMSE of 68.31241642. This is not as good as the SVM.

3.2 Final model

Now, we enter the last phase of our methodology: implementing the best model to predict on the values we left to the side at the start of the project.

3.2.1 Model choice

First, we will chose our model among the ones we tested.

```
model_names <- c("Naive prediction","Linear Model", "Linear Model with interaction terms", "Linear Model
rmse_scores <- c(rmse_0,rmse_1,rmse_2,rmse_3,rmse_4, rmse_5, rmse_6, rmse_7, rmse_8, rmse_9, rmse_10, rm
results_table <- data.frame(Model = model_names, RMSE = rmse_scores)
knitr::kable(results_table, row.names = FALSE)
```

Model	RMSE
Naive prediction	68.65346
Linear Model	68.56824
Linear Model with interaction terms	74.57194
Linear Model with top terms	67.31247
Linear Model with initial variable and top interactions	68.62204
Multinomial regression	68.55832
Multinomial regression with interaction terms	82.40140
Multinomial regression with top interaction terms	67.31247
Support Vector Regression	66.58567
Optimized Support Vector Regression	66.49303
Random forest regression	67.34369
ANN regression	72.35851

The best model we had was the Optimized Support Vector Regression at 66.49303 RMSE. We will reproduce it on our full train dataset and use the final test data to get results.

3.2.2 Final test data transformation

We need to apply the same transformations to the data we will use for the final test than what we did to the more regular.

```
encoded_final_test_set <- final_test_set
#Binaries
```

```r
encoded_final_test_set <- mutate(encoded_final_test_set, Gender = if_else(Gender=="Female", 1, 0))
encoded_final_test_set <- mutate(encoded_final_test_set, Family_History_of_OCD  = if_else(Family_History
encoded_final_test_set <- mutate(encoded_final_test_set, Depression_Diagnosis  = if_else(Depression_Dia
encoded_final_test_set <- mutate(encoded_final_test_set, Anxiety_Diagnosis  = if_else(Anxiety_Diagnosis
#Nominals

for(index in 1:length(Ethnicities)){
    value <- Ethnicities[index]
    name = paste("Ethnicity", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Ethnicity,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}
for(index in 1:length(Marital_Statuses)){
    value <- Marital_Statuses[index]
    name = paste("Marital_Status", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Marital_Status,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}
for(index in 1:length(Previous_Diagnoses)){
    value <- Previous_Diagnoses[index]
    name = paste("Previous_Diagnosis", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Previous_Diagnoses,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}
for(index in 1:length(Obsession_Types)){
    value <- Obsession_Types[index]
    name = paste("Obsession_Type", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Obsession_Type,
      value
```

```
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}
for(index in 1:length(Compulsion_Types)){
    value <- Compulsion_Types[index]
    name = paste("Compulsion_Type", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Compulsion_Type,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}
for(index in 1:length(Medications)){
    value <- Medications[index]
    name = paste("Medications", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Medications,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}
for(index in 1:length(Education_Level)){
    value <- Education_Level[index]
    name = paste("Education_Level", value, sep = "_")
    encoded_final_test_set[name] <- NA
    detection <-as.integer(
    str_detect(
      encoded_final_test_set$Education_Level,
      value
      )
    )
    detection[is.na(detection)] <- 0
    encoded_final_test_set[name] <- detection
}

names(encoded_final_test_set)[names(encoded_final_test_set) == "Previous_Diagnosis_Panic Disorder"] <-
names(encoded_final_test_set)[names(encoded_final_test_set) == "Obsession_Type_Harm-related"] <- "Obsess
names(encoded_final_test_set)[names(encoded_final_test_set) == "Education_Level_High School"] <- "Educa
names(encoded_final_test_set)[names(encoded_final_test_set) == "Education_Level_Some College"] <- "Educa
names(encoded_final_test_set)[names(encoded_final_test_set) == "Education_Level_College Degree"] <- "Edu
names(encoded_final_test_set)[names(encoded_final_test_set) == "Education_Level_Graduate Degree"] <- "E
```

```
encoded_final_test_set <- encoded_final_test_set[,c(-1, -4,-5,-6,-9,-11,-12,-17)]
```

```
#apply the scale
scales_final_test_set <- predict(scale, encoded_final_test_set)

#Separate duration of symptoms from the rest of the set
final_prediction_results <- scales_final_test_set[,4]
scaled_final_test_set<- scales_final_test_set[,-4]
```

We now have our three sets of data : - The scaled_work_set which will be used for training. - The scaled_final_test_set which will be used to predict the results - The final_prediction_results which will be used to verify the performance of our model for the prediction

3.2.3 Final model

```
svm_model <-svm(
  Duration_of_Symptoms~.,
  data = scaled_work_set,
  gamma = 0.01,
  cost = 1)
```

```
svm_prediction <- predict(
  svm_model,
  newdata = scaled_final_test_set
)
```

```
postResample(pred  = svm_prediction, obs  = final_prediction_results)
```

```
##         RMSE     Rsquared          MAE
## 71.818874496  0.001060598 61.648355908
```

```
rmse <- postResample(pred  = svm_prediction, obs  = final_prediction_results)[1]
```

Our RMSE dropped to 71.818874496 The tests had an RMSE of 66.49303.

4 Conclusion

We were working with a dataset of OCD cases, trying to predict the duration in month of the symptoms. After preparing the data, we compared several models, and chose to use a support vector model. During the trials, this gave us an RMSE of 66.49303. When we applied that model to the full dataset, we obtained an RMSE of 71.818874496.

Considering that our unit of length was the month, this means our errors are around 6 years, or about 30% of our full range of values. This is not a good prediction. As noted earlier, the Rsquared is also very low, 0.001060598 which means that the information we imputed explain 10% of the values.

If we consider the graph from earlier, we can confirm our suspicions: the lack of correlations between our variables makes an accurate prediction very difficult.

```
corrplot(cor(subset_data)[1,, drop=FALSE], method = "color", cl.pos='n', tl.col="black", tl.srt=45,tl.c
```

To conclude, we have a model, wich is better than giving our patient the average over all patients, since our RMSE beats the RMSE of our model beats the naive value by a few months. However it is still very far from being actually good.

If we were to expand this project in a broader context, such as a tool meant to be used in a real situation, these results would not be acceptable for deployment and we would need to pursue more steps to improve it. Considering that all variables had very small correlations with the duration of symptoms, our first goal would be to search for more variables that might explain a greater part of the phenomenon. In this hypothetical situation, we would have medical specialists as part of the teams, and we would need to consult them.