PRÀCTICA2: Neteja i anàlisi de les dades

Paula Sobrevals, Jordi Serra

05/01/2021

1. Integració i selecció de les dades d'interès a analitzar:

Em escollit el següent repositori per que conté diferents valors i tipologies, a l'hora que disposa de dos fitxers i ens permet crear un únic dataset a partir de fusionar els dos. Obtenim les dades del repositori: https://www.kaggle.com/lantanacamara/hong-kong-horse-racing Aquest conté dades dels resultats de les 1561 curses de cavalls fetes a Hong Kong entre el 14 de Setembre del 2014 i el 16 de Juliol de 2017. Està organitzat amb dos fitxers: race-result-horse.csv i race-result-race.csv:

```
horse.data <- read.csv("./race-result-horse.csv")
race.data <- read.csv("./race-result-race.csv")
```

Unifiquem els dos datasets en un a partir de la variable comú als dos fitxers. Aquesta fa referencia al identificador de la carrera: **race_id** i mostrem la dimensió de les observacions i les variables.

```
dataset <- merge(horse.data, race.data, by = "race_id")
dim(dataset)
```

```
## [1] 30189    30
```

2. Neteja de les dades:

Abans de netejar les dades, hem de mirar què hi tenim a cada columna. Per fer-ho utilizarem la funció summary() que ens permet visualitzar i fer-nos una idea general general el dataset:

```
#head(dataset,5)
str(dataset)
```

```
## 'data.frame':    30189 obs. of  30 variables:
##  $ race_id              : chr  "2014-001" "2014-001" "2014-001" "2014-001" ...
##  $ finishing_position   : chr  "1" "2" "3" "4" ...
##  $ horse_number         : int  1 2 10 3 7 9 13 4 6 11 ...
##  $ horse_name           : chr  "DOUBLE DRAGON" "PLAIN BLUE BANNER" "GOLDWEAVER" "SUPREME P
ROFIT" ...
##  $ horse_id             : chr  "K019" "S070" "P072" "P230" ...
##  $ jockey               : chr  "B Prebble" "D Whyte" "Y T Cheng" "J Moreira" ...
##  $ trainer              : chr  "D Cruz" "D E Ferraris" "Y S Tsui" "C S Shum" ...
##  $ actual_weight        : chr  "133" "133" "121" "132" ...
##  $ declared_horse_weight: chr  "1032" "1075" "1065" "1222" ...
##  $ draw                 : chr  "1" "13" "3" "2" ...
##  $ length_behind_winner : chr  "-" "2" "2" "2" ...
##  $ running_position_1   : int  1 8 2 6 9 12 4 5 7 11 ...
##  $ running_position_2   : int  2 9 1 4 10 13 3 6 7 11 ...
##  $ running_position_3   : int  2 9 1 5 10 13 3 6 7 12 ...
##  $ running_position_4   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ finish_time          : chr  "1.22.33" "1.22.65" "1.22.66" "1.22.66" ...
##  $ win_odds             : chr  "3.8" "8" "5.7" "6.1" ...
```

```
##  $ running_position_5   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ running_position_6   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ src                  : chr  "20140914-1.html" "20140914-1.html" "20140914-1.html" "2014
0914-1.html" ...
##  $ race_date            : chr  "14/9/14" "14/9/14" "14/9/14" "14/9/14" ...
##  $ race_course          : chr  "Sha Tin" "Sha Tin" "Sha Tin" "Sha Tin" ...
##  $ race_number          : int  1 1 1 1 1 1 1 1 1 ...
##  $ race_class           : chr  "Class 5" "Class 5" "Class 5" "Class 5" ...
##  $ race_distance        : int  1400 1400 1400 1400 1400 1400 1400 1400 1400 1400 ...
##  $ track_condition      : chr  "GOOD TO FIRM" "GOOD TO FIRM" "GOOD TO FIRM" "GOOD TO FIRM"
 ...
##  $ race_name            : chr  "TIM WA HANDICAP" "TIM WA HANDICAP" "TIM WA HANDICAP" "TIM
WA HANDICAP" ...
##  $ track                : chr  "TURF - \"A\" COURSE" "TURF - \"A\" COURSE" "TURF - \"A\" C
OURSE" "TURF - \"A\" COURSE" ...
##  $ sectional_time       : chr  "13.59 22.08 23.11 23.55" "13.59 22.08 23.11 23.55" "13.59
22.08 23.11 23.55" "13.59 22.08 23.11 23.55" ...
##  $ incident_report      : chr  "\n                When about to enter the track, SHANTARAA
M became fractious, reared on two occasions and then "| __truncated__ "\n                When
 about to enter the track, SHANTARAAM became fractious, reared on two occasions and then "| _
_truncated__ "\n                When about to enter the track, SHANTARAAM became fractious, r
eared on two occasions and then "| __truncated__ "\n                When about to enter the t
rack, SHANTARAAM became fractious, reared on two occasions and then "| __truncated__ ...
```

```
paste("Nombre de variables:" ,length(colnames(dataset)))
```

```
## [1] "Nombre de variables: 30"
```

Com podem veure, tenim 30 variables amb 30189 observacions. Com que hi ha moltes variables que estan repetides o que no aporten una informació vàlida, n'eliminarem algunes:

1. **incident_report**: Són comentaris sobre la cursa, no utilitzarem aquesta variable per als nostres anàlisis, per tant els podem eliminar.
2. **running_position_X**: Entenem que són les posicions intermitjes de la carrera. No ens aporta informació sobre abans de la carrera, sinó durant aquesta.
3. **SRC**: Tampoc ens aporta res més que el nom dels fitxers de dades.
4. **horse_number**: Depèn de cada cursa i no identifica el cavall, per tant deixem només el **horse_name**.
5. **race_number**: També és reduntant amb les altres variables.
6. **race_date**: La data de la carrera és indiferent per l'anàlisi de les dades.
7. **sectional_time**: Tindrem en compte només el temps total de la carrera, no de cada volta.
8. **horse_id** : Ja tenim un identificador pel cavall, el nom, per tant aquest és redundant.
9. **length_behind_winner** : És redundant, ja tenim la posició final de la cursa.

```
eliminar <- c("running_position_1","running_position_2","running_position_3","running_positio
n_4","running_position_5","running_position_6","incident_report","src","race_number","horse_n
umber","horse_id","sectional_time","length_behind_winner")
dataset_clean <- dataset[ , !(names(dataset) %in% eliminar)]
dataset_clean[] <- lapply(dataset_clean, as.character)
attach(dataset_clean) # Per poder fer referencia directament a les variables
```

2.1. Les dades contenen zeros o elements buits? Com gestionaries aquests casos?

Fent una ullada al dataset, veiem que alguns valors *NA* estan representats com a "—", "N", "SH" o "HD". Canviarem aquests valors a *NA* per a poder treballar amb la mateixa notació. A més a més, a la columna de **length_behind_winner** (la distància del cavall en qüestió en relació al primer), ens trobem valors representats com a "-".

Aquests corresponen al guanyador de la cursa, no té distància sobre sí mateix, és a dir, no són valors buits sinó que representen distància 0. Modificarem la notació per “0”. A la variable **finishing_position**, tenim diversos valors no numèrics, aquests estan representats amb diferents lletres. Per a utilitzar la mateixa notació, canviarem la variable a numèrica i aquells valors no numèrics passaran a estar representats amb “NA”.

```
dataset_clean[dataset_clean == "-"] <- "0"
dataset_clean[dataset_clean == "---"] <- NA
dataset_clean[dataset_clean == "N"] <- NA
dataset_clean[dataset_clean == "SH"] <- NA
dataset_clean[dataset_clean == "HD"] <- NA
dataset_clean$finishing_position  <- lapply(dataset_clean$finishing_position, as.character)
dataset_clean$finishing_position <- unlist(lapply(dataset_clean$finishing_position, as.intege
r)) # tots aquells valors que no s'hagin pogut transformar són els no numèrics, i restaran co
m a "NA".
```

Un cop ja tenim tots els valors buits amb la mateixa identificació, podem mirar si tenim gaires elements buits, i en quines variables.

```
#paste("Nombre de valors nulls:", sum(is.na(race_id)))
colSums(is.na(dataset_clean))
```

```
##               race_id    finishing_position            horse_name
##                     0                   825                     0
##                jockey               trainer         actual_weight
##                    29                     0                     0
## declared_horse_weight                  draw           finish_time
##                     0                   591                   669
##              win_odds             race_date           race_course
##                   591                     0                     0
##            race_class         race_distance       track_condition
##                     0                     0                     0
##             race_name                 track
##                     0                     0
```

Podem observar que tenim alguns valors buits al nostre dataset. Primer ens centrarem en la variable **finishing_position**. Determinem quin % de valors buits representa, per a poder decidir com actuem:

```
paste("Proporció de valors nulls de la variable finishing_position:", format((sum(is.na(datas
et_clean$finishing_position))/nrow(dataset_clean))*100, format = "f", digits = 3), "%")
```

```
## [1] "Proporció de valors nulls de la variable finishing_position: 2.73 %"
```

Veiem que representa un valor molt baix de totes les nostres dades, així que aquest cas el podem gestionar eliminant les observacions nules. Si representés una proporció més gran de les nostres dades, al voltant de 15-20% hauriem d’imputar els missing values, utilitzant un anàlisi de regressió. Observem quans valors nuls tenim ara:

```
dataset_clean <- dataset_clean[!(is.na(dataset_clean$finishing_position)),] # eliminem les fi
les sense resultats.
colSums(is.na(dataset_clean))
```

```
##               race_id    finishing_position            horse_name
##                     0                     0                     0
##                jockey               trainer         actual_weight
##                     0                     0                     0
## declared_horse_weight                  draw           finish_time
##                     0                     0                     0
```

```
##              win_odds             race_date           race_course
##                     0                     0                     0
##            race_class         race_distance       track_condition
##                     0                     0                     0
##             race_name                 track
##                     0                     0
```

Un cop netejada la variable, ja no tenim més valors buits en les altres columnes. Tot i així, si mirem al dataset original, veiem que hi havia bastants valors buits. Aquests, estaven concentrats en les columnes que hem eliminat perque no ens aportaven informació útil.

```
#summary(dataset[,-30])
colSums(is.na(dataset[,eliminar]))
```

```
##   running_position_1   running_position_2   running_position_3
##                  615                  629                  647
##   running_position_4   running_position_5   running_position_6
##                13571                26425                29640
##      incident_report                  src          race_number
##                    0                    0                    0
##         horse_number             horse_id       sectional_time
##                  338                    0                    0
## length_behind_winner
##                    0
```

2.2. Identificació i tractament de valors extrems

Per a visualitzar els valors extrems, primer escollirem aquelles variables en les quals podem trobar-nos valors extrems. De totes les variables del nostre dataset, només les variables continues poden tenir valors extrems, aquelles que categòriques o identificatives no en tindran (**race_id**, **horse_name**, **jockey**, **trainer**, **draw**, **race_course**, **race_name**, **race_class**, **track**, **track_condition**, **finishing_position**, **length_behind_winner**, **race_distance**). Primer transformem les variables continues a numèriques per poder trobar-ne els outliers.

```
dataset_clean$actual_weight <- unlist(lapply(dataset_clean$actual_weight, as.numeric))
dataset_clean$declared_horse_weight <- unlist(lapply(dataset_clean$declared_horse_weight, as.
numeric))
dataset_clean$finish_time <-  period_to_seconds(ms(dataset_clean$finish_time))
dataset_clean$win_odds <- unlist(lapply(dataset_clean$win_odds, as.numeric))
dataset_clean$race_distance <- as.factor(dataset_clean$race_distance)
```

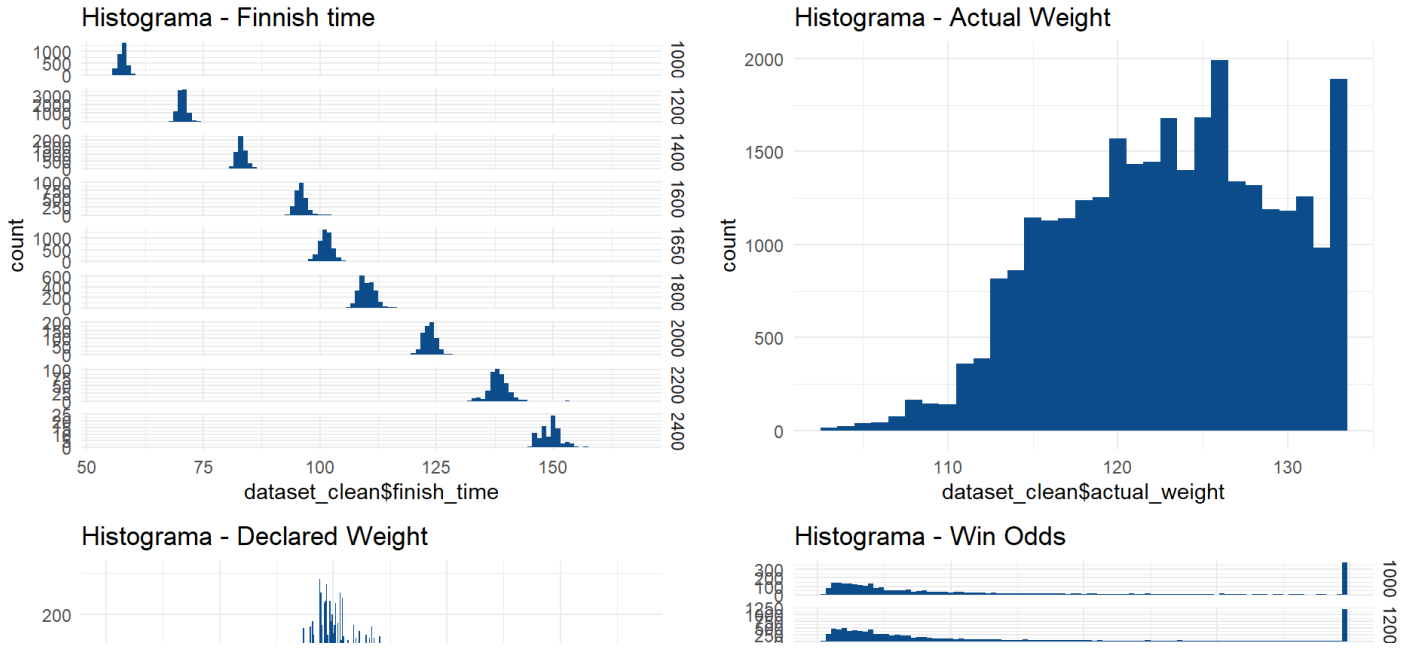
> ### Visualitzem els outliers:

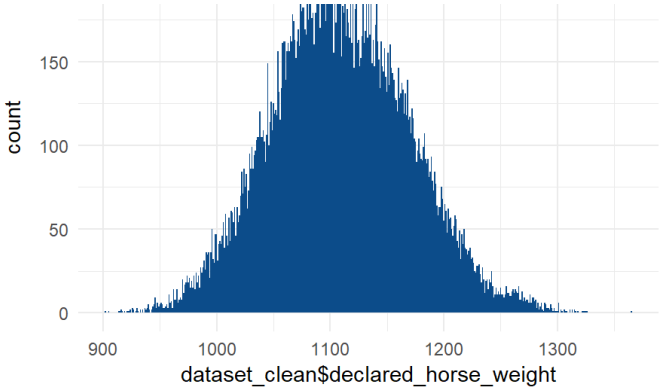
En el nostre dataset, hi tenim diverses curses de cavalls, aquestes tenen diferents distàncies de 1000m a 2400m. Segons amb quines variables hem de determinar els outliers tenint en compte la distància de la cursa, sinó aquest factor crearà un bias en els nostres resultats, si els tractem tots a la vegada quedarien amagats valors extrems dins de les longituds d'altres curses.

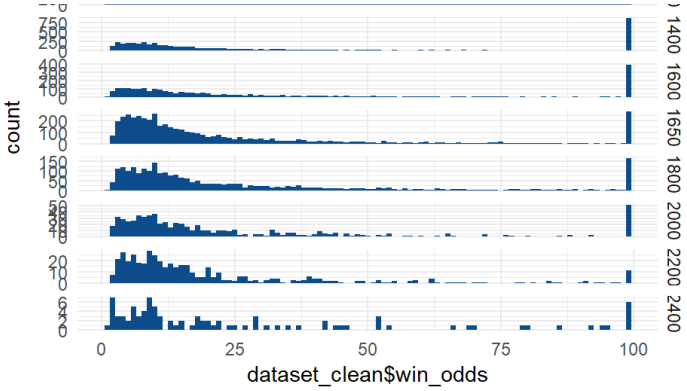
Utilitzem la representació gràfica (histograma i boxplot) per a visualitzar i detectar els outliers:

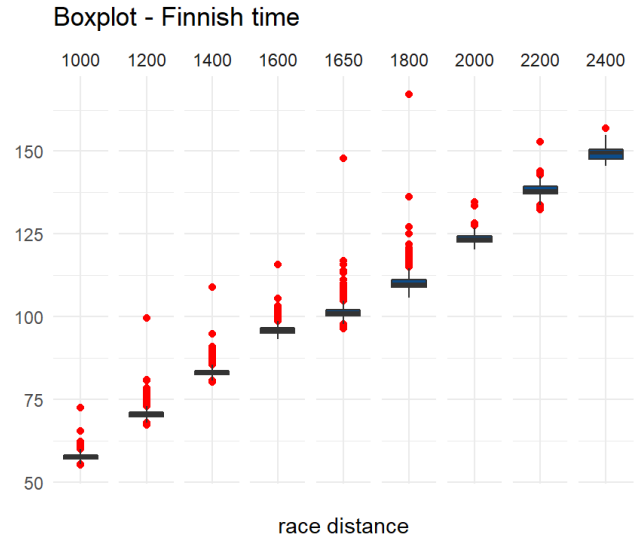
```
hist_f <- ggplot(data = dataset_clean, aes(x=dataset_clean$finish_time)) +
  facet_grid(rows = vars(dataset_clean$race_distance), scale = "free_y") +
  geom_histogram(binwidth=1, fill="#0c4c8a") +
  theme_minimal() +
  ggtitle("Histograma - Finnish time")
hist_aw <- ggplot(data = dataset_clean, aes(x=dataset_clean$actual_weight)) +
```

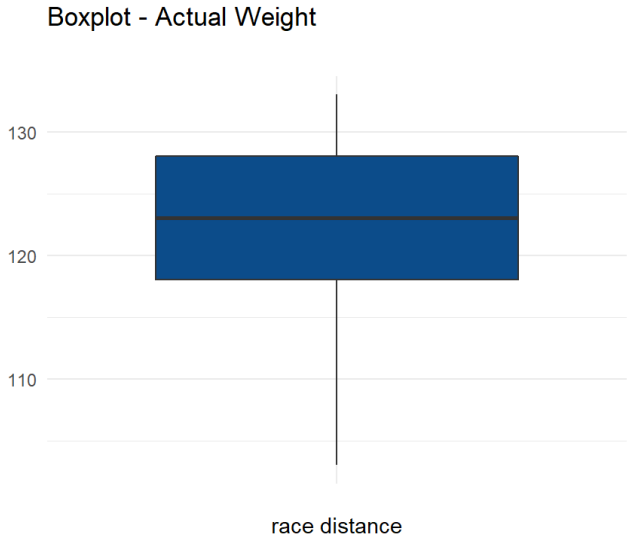
```
  geom_histogram(binwidth=1, fill="#0c4c8a") +
  theme_minimal() +
  ggtitle("Histograma - Actual Weight")
hist_dw <- ggplot(data = dataset_clean, aes(x=dataset_clean$declared_horse_weight)) +
  geom_histogram(binwidth=1, fill="#0c4c8a") +
  theme_minimal() +
  ggtitle("Histograma - Declared Weight")
hist_wo <- ggplot(data = dataset_clean, aes(x=dataset_clean$win_odds)) +
  facet_grid(rows = vars(dataset_clean$race_distance), scale = "free_y") +
  geom_histogram(binwidth=1, fill="#0c4c8a") +
  theme_minimal() +
  ggtitle("Histograma - Win Odds")
box_f <- ggplot(dataset_clean) +
  aes(x = "", y = dataset_clean$finish_time) +
  facet_grid(cols = vars(dataset_clean$race_distance), scale = "free") +
  geom_boxplot(fill = "#0c4c8a", outlier.colour = "red") +
  ggtitle("Boxplot - Finnish time") +
  ylab("") + xlab("race distance") +
  theme_minimal()
box_aw <- ggplot(dataset_clean) +
  aes(x = "", y = dataset_clean$actual_weight) +
  geom_boxplot(fill = "#0c4c8a",outlier.colour = "red") +
  ggtitle("Boxplot - Actual Weight") +
  ylab("") + xlab("race distance") +
  theme_minimal()
box_dw <- ggplot(dataset_clean) +
  aes(x = "", y = dataset_clean$declared_horse_weight) +
  geom_boxplot(fill = "#0c4c8a",outlier.colour = "red") +
  ggtitle("Boxplot - Declared weight") +
  ylab("") + xlab("race distance") +
  theme_minimal()
box_wo <- ggplot(dataset_clean) +
  aes(x = "", y = dataset_clean$win_odds) +
  facet_grid(cols = vars(dataset_clean$race_distance), scale = "free") +
  geom_boxplot(fill = "#0c4c8a",outlier.colour = "red") +
  ggtitle("Boxplot - Win Odds") +
  ylab("") + xlab("race distance") +
  theme_minimal()
hist_f + hist_aw + hist_dw + hist_wo + box_f + box_aw + box_dw + box_wo + plot_layout(nrow =
4, ncol = 2)
```

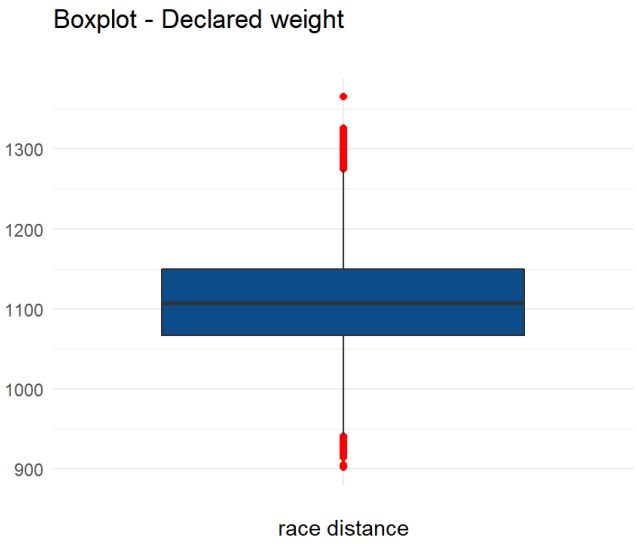


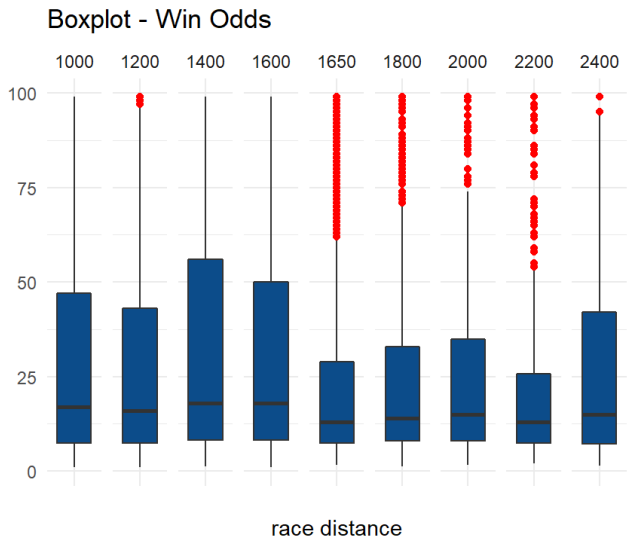












Per la variable del temps de finalització de cada carrera (*Finish_time*), podem veure que hi ha cavalls que triguem especialment molt més que la resta en acabar la cursa. Podem veure també, com el pes del genet (*Actual_weight*) està esbiaixat a l'alça, però tot i així hi ha alguns valors baixos, però no estan considerats outliers. El mateix passa amb el pes dels cavalls (*Declared_weight*), que la majoria estan distribuïts centralment, però tot i això hi ha valors que queden marcats com a extrems. Podem veure que la variació de pes és molt petita i sembla que els cavalls més grans són els que fan curses més curtes, però no és significatiu. Pel que fa als *win_odds*, veiem que tot i la mitjana ser bastant baixa, tenim molts valors alts, els quals estan qualificats com a extrems.

> ## Ara eliminem els outliers de cada una de les variables:

Eliminarem tots aquells valors outliers. Tot i així, podem observar que a la variable *win_odds*, tots els outliers són els cavalls que tenen més probabilitats de guanyar. Com que el nostre estudi té a veure amb aquests, no eliminarem aquests outliers del nostre dataset i els tindrem en compte pels anàlisis posteriors.

```
out_aw <- unlist(ggplot_build(box_aw)[["data"]][[1]][["outliers"]])
ind_aw <- which(dataset_clean$actual_weight %in% c(out_aw))
out_f <- unlist(ggplot_build(box_f)[["data"]][[1]][["outliers"]])
ind_f <- which(dataset_clean$finish_time %in% c(out_f))
out_dw <- unlist(ggplot_build(box_dw)[["data"]][[1]][["outliers"]])
ind_dw <- which(dataset_clean$declared_horse_weight %in% c(out_dw))
dataset_outliers <- dataset_clean[-c(ind_f,ind_dw,ind_aw),]
paste("Hem eliminat", (nrow(dataset_clean)-nrow(dataset_outliers)) , "outliers.")
```

```
## [1] "Hem eliminat 1277 outliers."
```

```
paste("Ara el nostre dataset té", nrow(dataset_outliers), "observacions.")
```

```
## [1] "Ara el nostre dataset té 28087 observacions."
```

```
dataset_clean <- dataset_outliers
dataset_clean <- dataset_clean[ , !(names(dataset_clean) %in% c("race_date"))]
```

3. Anàlisi de les dades.

Anem a fer ara l'anàlisi de les dades en profunditat.

3.1. Selecció dels grups de dades que es volen analitzar/comparar (planificació dels anàlisis a aplicar).

Visualitzem el datset tal i com l'hem deixat després de la neteja:

```
str(dataset_clean)
```

```
## 'data.frame':    28087 obs. of  16 variables:
##  $ race_id              : chr  "2014-001" "2014-001" "2014-001" "2014-001" ...
##  $ finishing_position   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ horse_name           : chr  "DOUBLE DRAGON" "PLAIN BLUE BANNER" "GOLDWEAVER" "SUPREME P
ROFIT" ...
##  $ jockey               : chr  "B Prebble" "D Whyte" "Y T Cheng" "J Moreira" ...
##  $ trainer              : chr  "D Cruz" "D E Ferraris" "Y S Tsui" "C S Shum" ...
##  $ actual_weight        : num  133 133 121 132 125 123 115 129 127 119 ...
##  $ declared_horse_weight: num  1032 1075 1065 1222 1136 ...
##  $ draw                 : chr  "1" "13" "3" "2" ...
##  $ finish_time          : num  82.3 82.7 82.7 82.7 83 ...
##  $ win_odds             : num  3.8 8 5.7 6.1 6.1 24 99 21 10 27 ...
##  $ race_course          : chr  "Sha Tin" "Sha Tin" "Sha Tin" "Sha Tin" ...
##  $ race_class           : chr  "Class 5" "Class 5" "Class 5" "Class 5" ...
##  $ race_distance        : Factor w/ 9 levels "1000","1200",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ track_condition      : chr  "GOOD TO FIRM" "GOOD TO FIRM" "GOOD TO FIRM" "GOOD TO FIRM"
 ...
##  $ race_name            : chr  "TIM WA HANDICAP" "TIM WA HANDICAP" "TIM WA HANDICAP" "TIM
WA HANDICAP" ...
##  $ track                : chr  "TURF - \"A\" COURSE" "TURF - \"A\" COURSE" "TURF - \"A\" C
OURSE" "TURF - \"A\" COURSE" ...
```

Tenim algunes variables numèriques contínues i d'altres categòriques. Treballarem amb totes les dades que tenim al *dataset_clean*, un cop hem eliminat els outliers i les variables repetides o que no eren útils. Separararem el dataset, i

treballarem per separat amb les dades numèriques continues i les dades categòriques, ja que per a treballar-les molts cops calen diferents funcions. Finalment per a dur a terme els anàlisis, juntarem el dataset i trasformarem aquelles variables char en numèriques, així podrem analitzar-lo tot sencer.

A més a més, les nostres dades estan mesurades amb diferents llargades de curses, aquest fet, altera els valors d'algunes variables (per exemple, el temps d'una cursa de 1000m serà molt diferent al d'una cursa de 2400m) per tant, per a analitzar la forma de les dades caldrà separar-les per llargada. Es a dir, si l'analisi ho requereix, farem l'estudi per separat de cada tipus de cursa per poder diferenciar bé el comportament. Com veurem per exemple en la regresió.

3.2. 1. Comprovació de la normalitat i homogeneïtat de la variància.

Anem a veure la normalitat i homogeneitat de la variança.

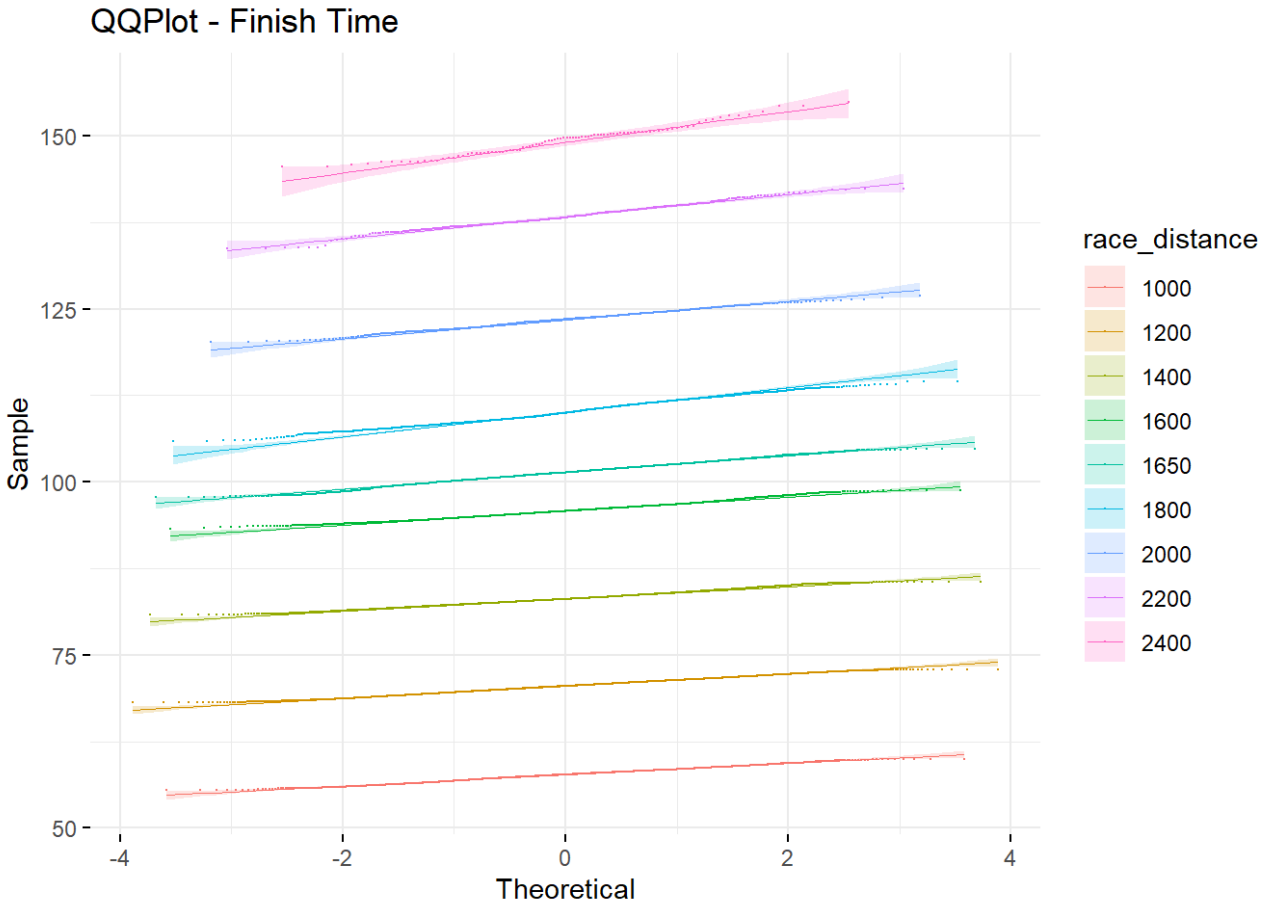
Per a comporvar la normalitat i la homogeneitat, mirarem com estan distribuides les dades, per a comprovar si compleixen amb la hipotesi de la distribució normal de les dades, i si es comporten de manera homogenia. Ho farem de dues maneres:

a. De manera visual: Amb un QQPlot, que ens permetrà veure si les dades s'ajusten a la normalitat, i amb un scatter plot que ens ajudarà a veure si es comporten de manera homogènia.

b. Utiltzant tests específics: Per comporvar la normalitat usarem el Test de Shapiro–Wilk, el qual és considerat un dels test més potents per a contrastar la normalitat. Per comporvar la homogeneitat, usarem el Lavene Test i el Fligner-Killeen test segons si les nostres dades segueixen una distribució normal o no.

1. Finish_time

```
ggqqplot(data=dataset_clean, x="finish_time", color = "race_distance", ggtheme = theme_minima
l(), conf.int = TRUE, size =0.1, title = "QQPlot - Finish Time")
```




```
SW <- list()
llargada = c(1000,1200,1400,1600,1800,2000,2200,2400)
i = 1
```

```
for (ll in llargada) {
  dades <- dataset_clean[dataset_clean$race_distance == ll,9]
  if (length(dades) < 5000) {
    SW[[i]] <- shapiro.test(dades)
    }
  else{
    SW[[i]] <- shapiro.test(dades[1:5000])
    }
  i = i+1
}
paste("La mitjana del valor W del Test de Shapiro–Wilk:" , mean(c(SW[[1]]$statistic[[1]],SW[[
2]]$statistic[[1]],SW[[2]]$statistic[[1]],SW[[2]]$statistic[[1]],SW[[2]]$statistic[[1]],SW[[2
]]$statistic[[1]],SW[[2]]$statistic[[1]],SW[[2]]$statistic[[1]])))
```

```
## [1] "La mitjana del valor W del Test de Shapiro–Wilk: 0.997544534505758"
```

```
paste("El p-value mitjà del Test de Shapiro–Wilk:", mean(c(SW[[1]]$p.value[[1]],SW[[2]]$p.val
ue[[1]],SW[[2]]$p.value[[1]],SW[[2]]$p.value[[1]],SW[[2]]$p.value[[1]],SW[[2]]$p.value[[1]],S
W[[2]]$p.value[[1]],SW[[2]]$p.value[[1]])))
```

```
## [1] "El p-value mitjà del Test de Shapiro–Wilk: 1.11693661605265e-06"
```

```
leveneTest(finish_time ~ as.factor(race_distance), data=dataset_clean)
```

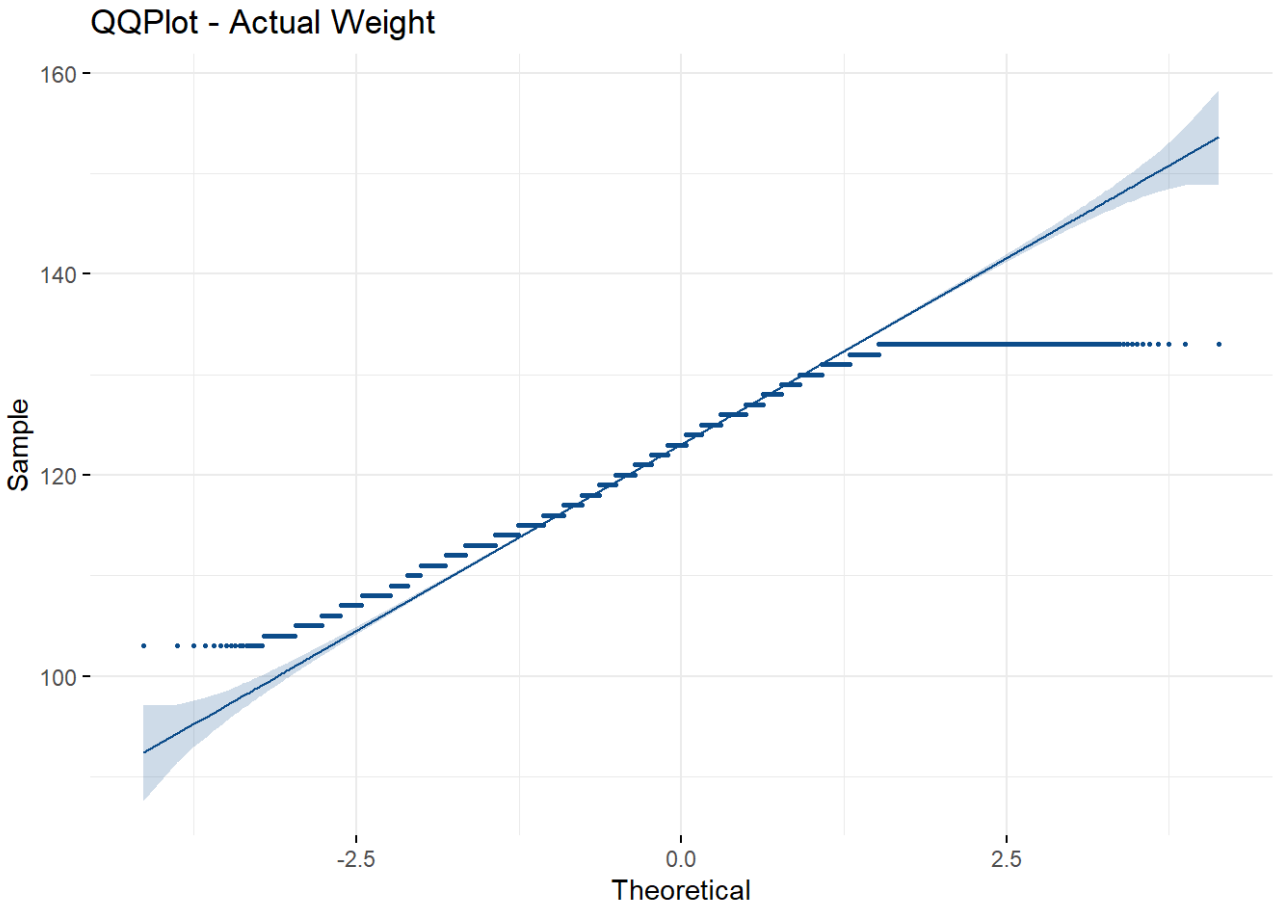
```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     8  366.78 < 2.2e-16 ***
##       28078
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Podem veure en el QQPlot que les dades de cada cursa s'ajusten molt bé a la normalitat.

A més el Test de Shapiro–Wilk també ens indica que les nostres dades són normalitzades, el *p-value* és molt significant i el valor de Shapiro (W) és quasi 1. Segons el Test de Levene, veiem que tenim un *p-value* inferior al nivell de significació (<0,05) per tant rebutjarem la hipòtesi nul·la d'homoscedasticitat i concluïm que la variable presenta variàncies estadísticament diferents per als diferents grups.

2. Actual Weight

```
dades <- sample(seq(1,nrow(dataset_clean), by =1), 5000)
ggqqplot(data=dataset_clean, x="actual_weight", ggtheme = theme_minimal(), col = "#0c4c8a", c
onf.int = TRUE, size = 0.5, title = "QQPlot - Actual Weight")
```




```
shapiro.test(dataset_clean$actual_weight[dades])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dataset_clean$actual_weight[dades]
## W = 0.97553, p-value < 2.2e-16
```

```
leveneTest(actual_weight ~ as.factor(race_distance), data=dataset_clean)
```

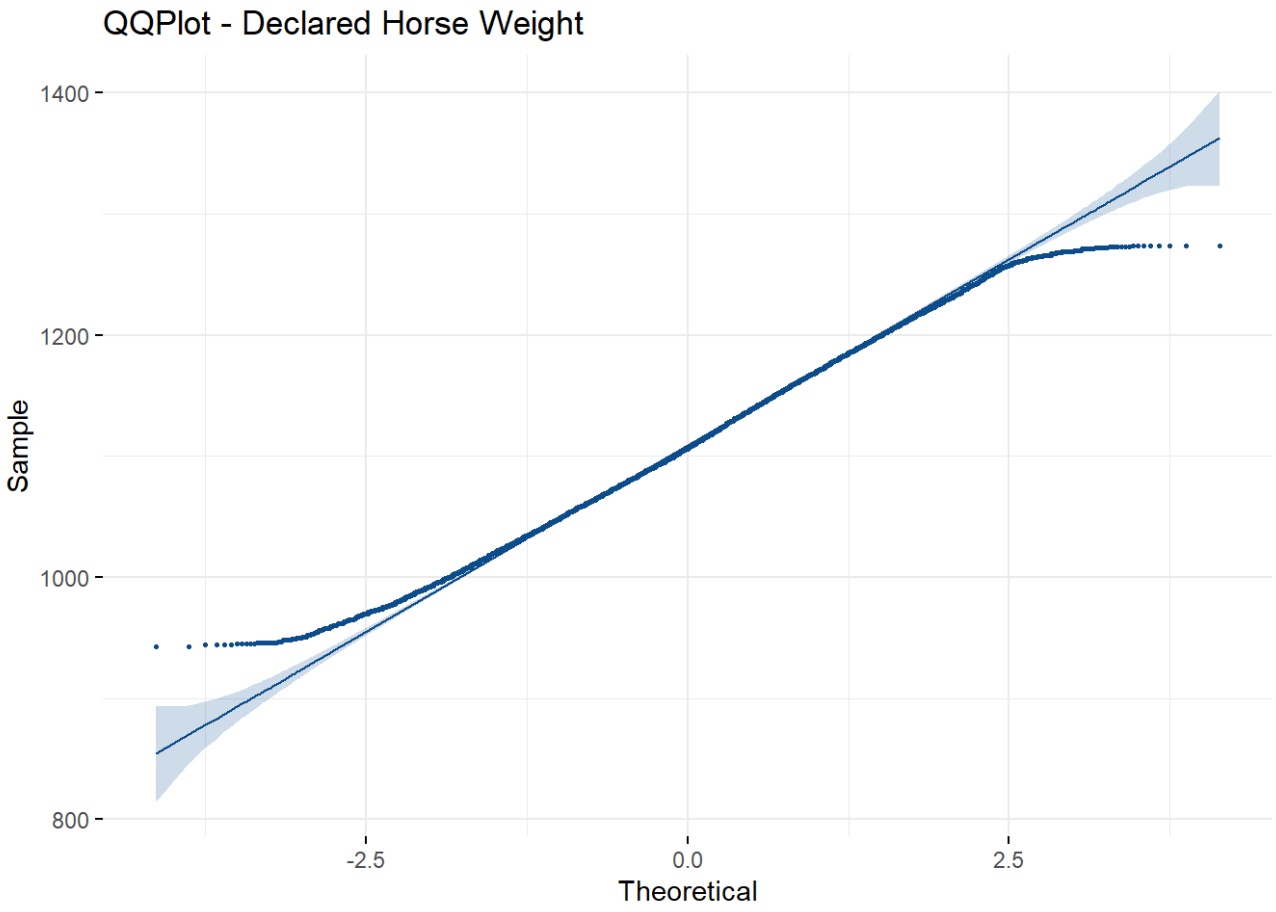
```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     8  18.558 < 2.2e-16 ***
##       28078
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

En aquets cas, al *qqplot* veiem que les dades segueixen la linia de la normalitat, tot i que cap al final s'estanquen al voltant de 132 jin (mesura de Xina ~= 65 kg).

Segons el Test de Shapiro–Wilk, veiem que les dades també estan normalitzades, el *p-value* és molt significant i el valor de Shapiro (W) és quasi 1. Segons el Test de Levene, veiem que tenim un *p-value* inferior al nivell de significació (<0,05) per tant rebutjarem la hipòtesi nul·la d'homoscedasticitat i concluïm que la variable presenta variàncies estadísticament diferents per als diferents grups.

3. Declared Horse Weight

```
dades <- sample(seq(1,nrow(dataset_clean), by =1), 5000)
ggqqplot(data=dataset_clean, x="declared_horse_weight", ggtheme = theme_minimal(), col = "#0c
4c8a", conf.int = TRUE, size = 0.5, title = "QQPlot - Declared Horse Weight")
```




```
shapiro.test(dataset_clean$declared_horse_weight[dades])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dataset_clean$declared_horse_weight[dades]
## W = 0.99722, p-value = 5.863e-08
```

```
leveneTest(declared_horse_weight ~ as.factor(race_distance), data=dataset_clean)
```

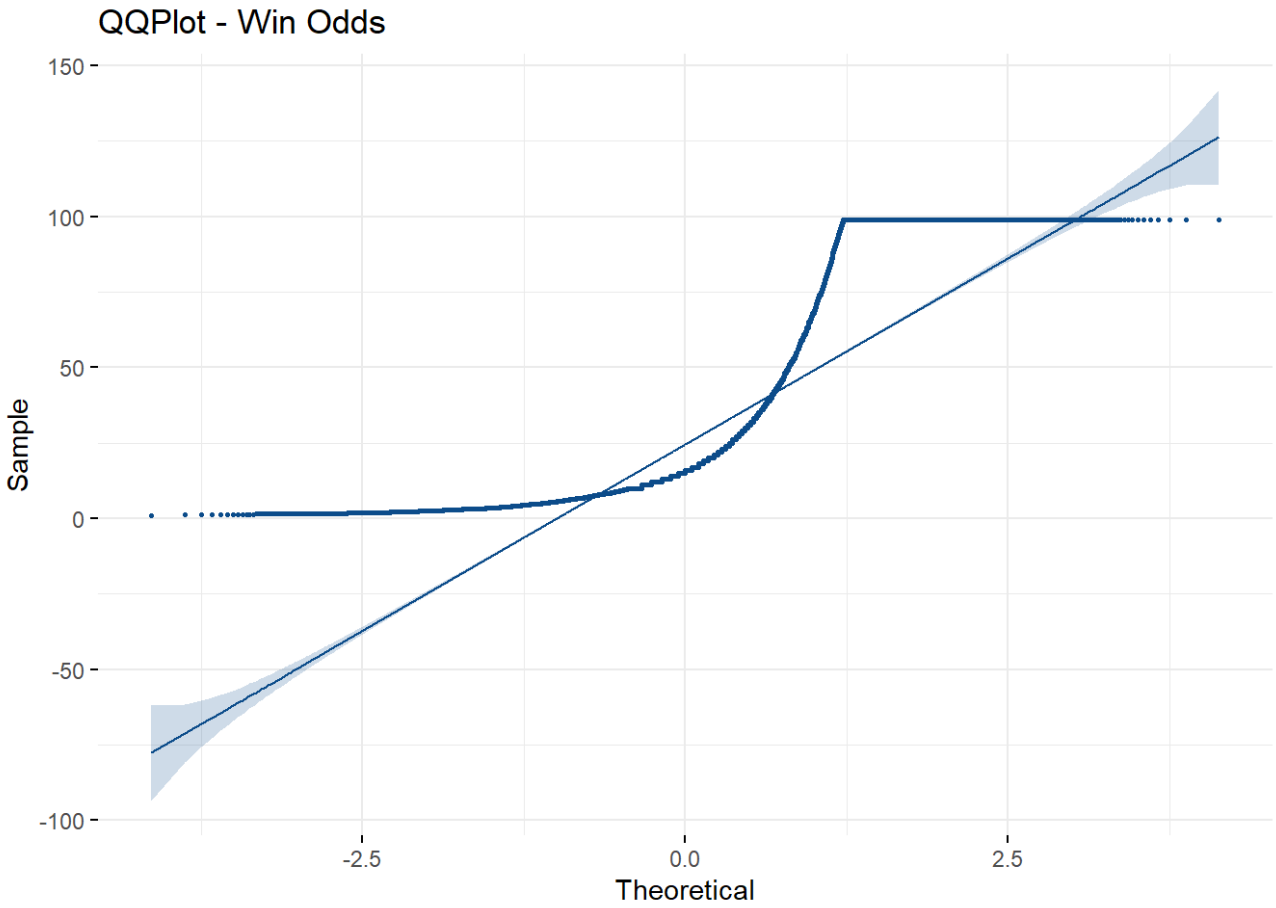
```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value  Pr(>F)
## group     8  2.2772 0.01968 *
##       28078
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

En aquets cas, al *qqplot* veiem que les dades s'ajusten molt bé a la linia de la normalitat.

Segons el Test de Shapiro–Wilk, veiem que les dades també estan normalitzades, el *p-value* és molt significant i el valor de Shapiro (W) és quasi 1. Segons el Test de Levene, veiem que tenim un *p-value* inferior al nivell de significació (<0,05) per tant rebutjarem la hipòtesi nul·la d'homoscedasticitat i concluïm que la variable presenta variàncies estadísticament diferents per als diferents grups.

4. Win Odds

```
dades <- sample(seq(1,nrow(dataset_clean), by =1), 5000)
ggqqplot(data=dataset_clean, x="win_odds", ggtheme = theme_minimal(), col = "#0c4c8a", conf.i
nt = TRUE, size = 0.5, title = "QQPlot - Win Odds")
```




```
shapiro.test(dataset_clean$win_odds[dades])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dataset_clean$win_odds[dades]
## W = 0.75102, p-value < 2.2e-16
```

```
fligner.test(win_odds ~ as.factor(race_distance), data=dataset_clean)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  win_odds by as.factor(race_distance)
## Fligner-Killeen:med chi-squared = 401.25, df = 8, p-value < 2.2e-16
```

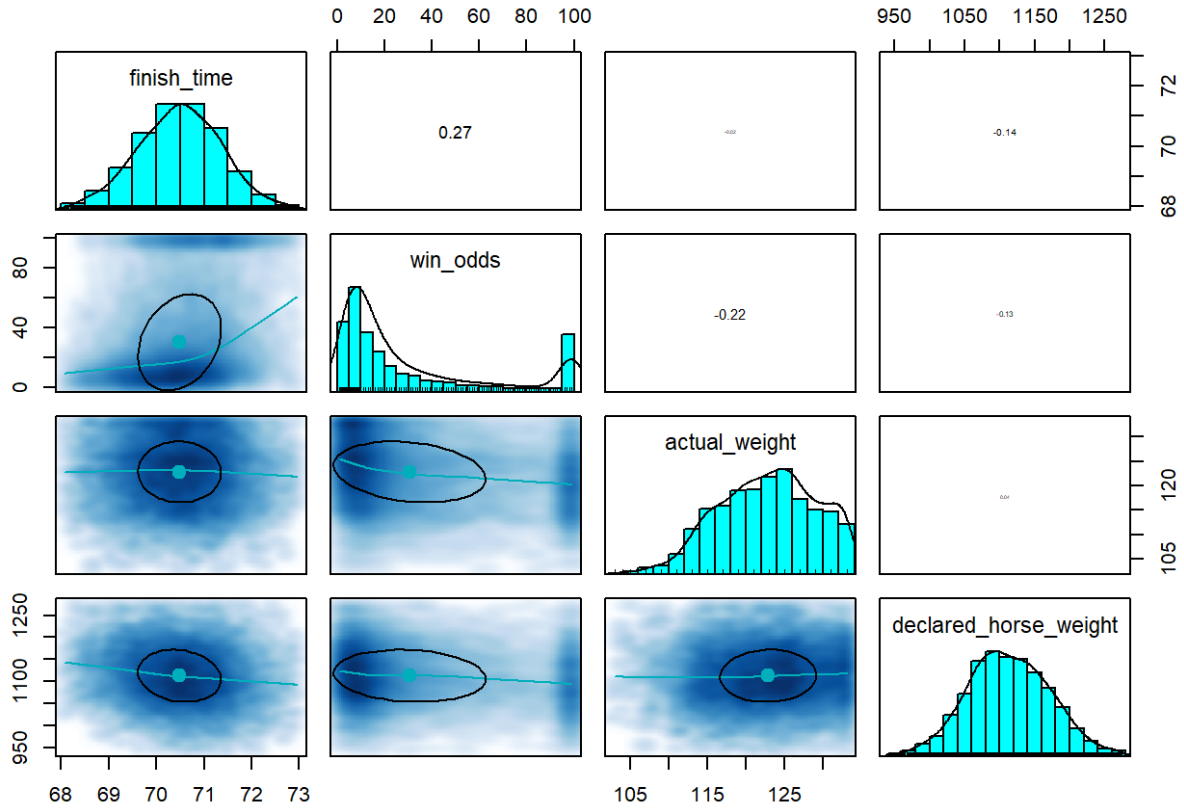
En aquest cas, les dades sembla que no segueixen una forma normal. Al *QQPlot*, podem observar que les porbabilitats de guanyar estan repartides de manera “biestable”, tant al 0% com al 100%, i molt poques entremig. Pel que fa al Test de Shapiro–Wilk, el valor (W) és molt més baix que els anteriors (0.76), però el *p-value* segueix sent significant. En aquest cas, utilitzem el Test de Fligner-Killeen, ja que la nostra variable no segueix una distribució normal. Tot i així, veiem que altre cop tenim un *p-value* inferior al nivell de significació (<0,05) per tant rebutjarem la hipòtesi nul·la d’homoscedasticitat i concluïm que la variable presenta variàncies estadísticament diferents per als diferents grups.

Homogeneitat:

Per a visualitzar la homogeneitat, separarem les dades segons la *race_distance* i utilitzarem la *race_distance* = 1200 com a mostra representativa, ja que és la més extensa (composta per ~9633 observacions). Veurem de manera senzilla que les variables numèriques, tal i com hem comprovat anteriorment amb els tests de Lavene i Fligner-Killeen, no són homogènies entre elles.

```
dataset_1200 <- dataset_clean[dataset_clean$race_distance == "1200",]

pp <- pairs.panels(dataset_1200[,c("finish_time","win_odds","actual_weight","declared_horse_weight")], method = "pearson", col = "#00AFBB",density = TRUE,  ellipses = TRUE,  show.points = TRUE, scale = TRUE, smoother = TRUE)
```




```
pp
```

```
## NULL
```

3.3. Aplicació de proves estadístiques per comparar els grups de dades. En funció de les dades i de l'objectiu de l'estudi, aplicar proves de contrast d'hipòtesis, correlacions, regressions, etc. Aplicar almenys tres mètodes d'anàlisi diferents.

1. Correlació:

Calcularem les correlacions de les variables, ja que aquestes ens poden ajudar a veure si alguna de les variables té més poder o està més relacionada amb qui guanyarà la cursa. Com que el nostre objectiu és acabar creant un model per a predir el resultat de la carrera, conèixer les correlacions ens pot ser molt útil per identificar aquelles variables que afecten més al resultat.

Pearson

Per a veure les correlacions entre les variables numèriques utilitzarem la correlació de Pearson:

Mirem la correlación entre els valors que són *numerics*. Com ja hem comentat anteriorment, aquestes no es comporten

igual i com podem veure hi ha una correlació directe entre el temps de la cursa i la distància. També sabem que el *finish_position* és una representació del *finish_time*, sense estar correlacionada amb la distància.

```
paste("Correlació del finish time amb la race distance:" , cor(x=dataset_clean$finish_time, y
=as.numeric(dataset_clean$race_distance)))
```

```
## [1] "Correlació del finish time amb la race distance: 0.992764688550992"
```

```
paste("Correlació del finish position amb la race distance:" ,cor(x=dataset_clean$finishing_p
osition, y=as.numeric(dataset_clean$race_distance)))
```

```
## [1] "Correlació del finish position amb la race distance: 0.00398758915056141"
```

Acabem de veure que el *finish_time* depèn de la *race_distance*, sembla logic que el temps estigui relacionat amb la llargada de la cursa, per tant, per a fer les correlacions no usarem aquesta variable. En aquest cas, la variable *finish_position* és una representació del *finish_time*, sense estar correlacionada amb el *race_distance*. Així doncs, usarem aquesta per a estudiar les correlacions entre la resta de variables:

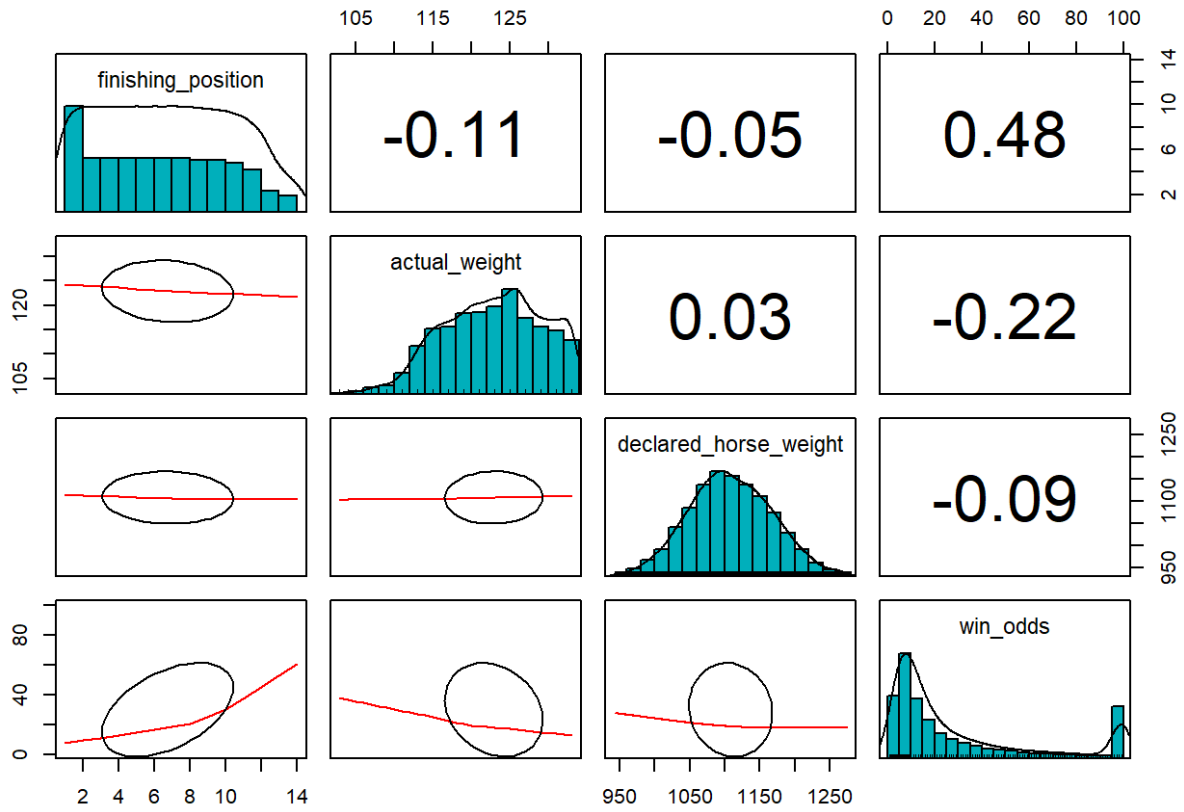
```
eliminar <- c("race_distance", "finish_time")
dataset_numerical <- dataset_clean[ , !(names(dataset_clean) %in% eliminar)]
dataset_numerical <- select_if(dataset_numerical, is.numeric)

cor <- rcorr(as.matrix(dataset_numerical), type = "pearson")
cor
```

```
##                       finishing_position actual_weight declared_horse_weight
## finishing_position                  1.00         -0.11                 -0.05
## actual_weight                      -0.11          1.00                  0.03
## declared_horse_weight              -0.05          0.03                  1.00
## win_odds                            0.48         -0.22                 -0.09
##                       win_odds
## finishing_position        0.48
## actual_weight            -0.22
## declared_horse_weight    -0.09
## win_odds                  1.00
##
## n= 28087
##
##
## P
##                       finishing_position actual_weight declared_horse_weight
## finishing_position                     0             0
## actual_weight          0                             0
## declared_horse_weight  0                 0
## win_odds               0                 0           0
##                       win_odds
## finishing_position     0
## actual_weight          0
## declared_horse_weight  0
## win_odds
```

```
pairs.panels(dataset_numerical,
             method = "pearson", # correlation method
             hist.col = "#00AFBB",
             density = TRUE,  # show density plots
```

```
            ellipses = TRUE, # show correlation ellipses
            cor = TRUE,
            show.points = FALSE
            )
```



Podem veure que no tenim correlacions entre les variables numèriques o són gairebé nules. A més a més, els *p-values* tots són significants (<0.05). No tenir correlacions ens indica que les variables entre elles són independents. És a dir, que podem analitzar-les sense trobar-nos biaixos de collinearitat. En el *pairplot*, veiem que quasi no hi ha homogeneitat entre les variables, tot i així *win_odds* presenta petites homogeneitats amb les variables: amb *finishing_position* hi veiem una homogeneitat positiva i amb *actual_weight* hi veiem una petita homogeneitat negativa.

ANOVA

Ara ens centrarem amb les correlacions entre variables categòriques, per a fer-ho utilitzarem la funció **ANOVA**, per a comparar les mitjanes entre més de dos grups de dades

```
dataset_categoric <- dataset_clean[ , !(names(dataset_clean) %in% c(names(dataset_numerical), "finish_time"))]
dataset_categoric[] <- lapply(dataset_categoric, as.factor)
dataset_categoric$finishing_position <- dataset_clean$finishing_position

res.aov <- aov(finishing_position ~ horse_name + jockey + trainer + draw + race_course + race_class + race_distance + track_condition + track, data = dataset_categoric)
summary(res.aov)
```

```
##                   Df Sum Sq Mean Sq F value   Pr(>F)
## horse_name      2125  85123    40.1   4.021  < 2e-16 ***
## jockey            83  16081   193.7  19.450  < 2e-16 ***
## trainer           23    925    40.2   4.038 2.44e-10 ***
## draw              14   5597   399.8  40.138  < 2e-16 ***
```

```
## race_course         1   2162  2161.5 216.994  < 2e-16 ***
## race_class         15  11219   747.9  75.083  < 2e-16 ***
## race_distance       8    572    71.5   7.179 1.52e-09 ***
## track_condition     8    123    15.4   1.542    0.137
## track               6    532    88.7   8.908 9.75e-10 ***
## Residuals       25803 257030    10.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

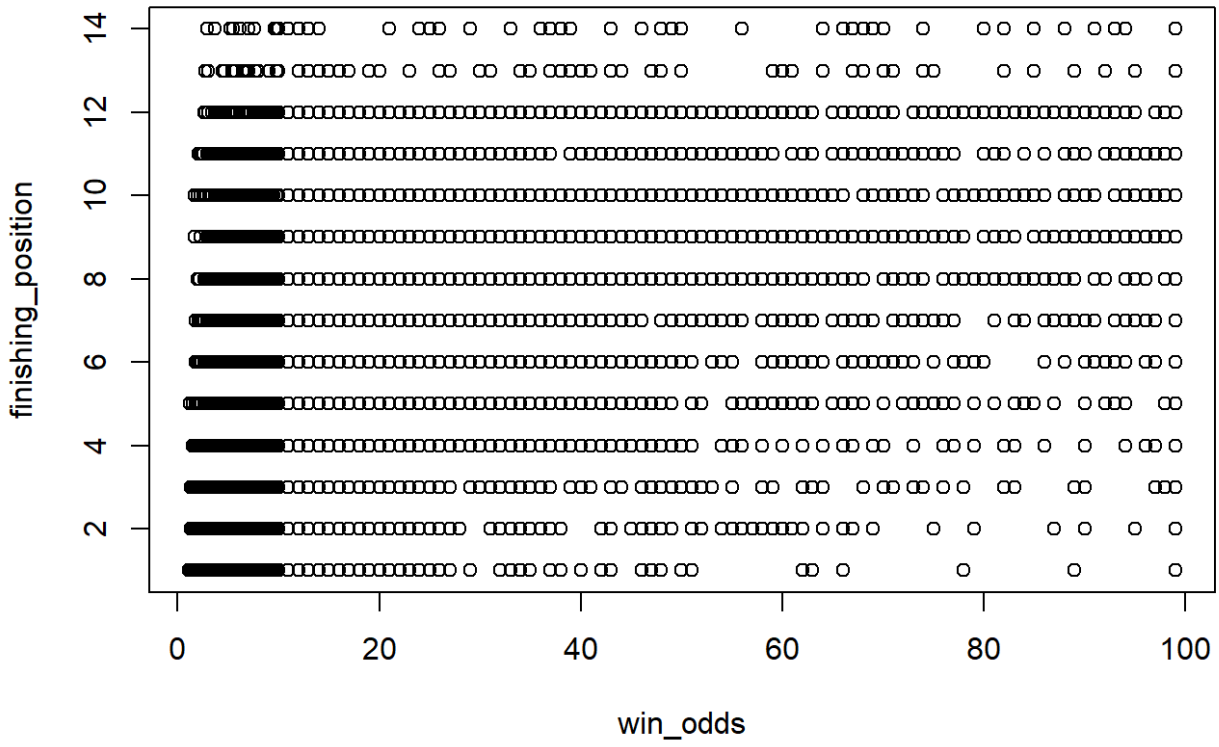
El resultat de la ANOVA té un *p-value* significant (<0.05) per totes les variables, excepte *track_condition*. És a dir, tal i com hem vist amb el pearson amb les variables numèriques, les categòriques també són independents. Pel que fa al *track_condition*, no hi ha una diferència significant sobre la variable *final_position* en relació al *track_condition* (Fvalue = 1.5).

2. Regressió:

Anem a comprovar ara la relació de dependencia entre les variables independent i la dependent que podem tenir en el nostre cas.

Aquí tenim com a variable depenent la posició final i la resta com a independent. Podem veure a la grafica següent que agafant una única variable no tenim una clara dependencia entre aquestes, però generant el model per cada un dels diferents tipus de cursa, tenint en compte la llargada d'aquesta, veiem com el valor de relació R-squared és molt alt, per sobre del 0.85, demostrant que hi ha una forta dependencia entre les variables i la posició final del cavall en cada una de les curses.

```
dades <- dataset_clean[dataset_clean$race_distance =="1200", ]
plot(finishing_position~win_odds,data=dades)
```




```
m1 <- lm(finishing_position~actual_weight+declared_horse_weight+win_odds+trainer+jockey+horse
_name+race_class+race_course+track_condition+race_name+race_id,data=dades)
summary(m1)
```

```
##
## Call:
## lm(formula = finishing_position ~ actual_weight + declared_horse_weight +
##     win_odds + trainer + jockey + horse_name + race_class + race_course +
##     track_condition + race_name + race_id, data = dades)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.851 -1.796  0.000  1.693 10.144
##
## Coefficients: (593 not defined because of singularities)
##
       Estimate
## (Intercept)
      -5.521307
## actual_weight
       0.092589
## declared_horse_weight
      -0.003926
## win_odds
       0.026669
## trainerA S Cruz
      -0.072254
## trainerA Schutz
      -1.437090
## ...
## ...
## race_id2016-797

## race_id2016-799

## race_id2016-806

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.985 on 7141 degrees of freedom
## Multiple R-squared:  0.4724, Adjusted R-squared:  0.2884
## F-statistic: 2.567 on 2491 and 7141 DF,  p-value: < 2.2e-16
```

3. PCA + Regressió:

Aquesta prova la farem per a comparar-la amb la regressió estàndard. Volem veure si podem millorar el model creat anteriorment amb la regressió, utilitzan els principal components necessaris com a predictors per a ajustar el model de manera més precisa.

Primer transformem els valors categòrics a numèrics per a poder fer el PCA:

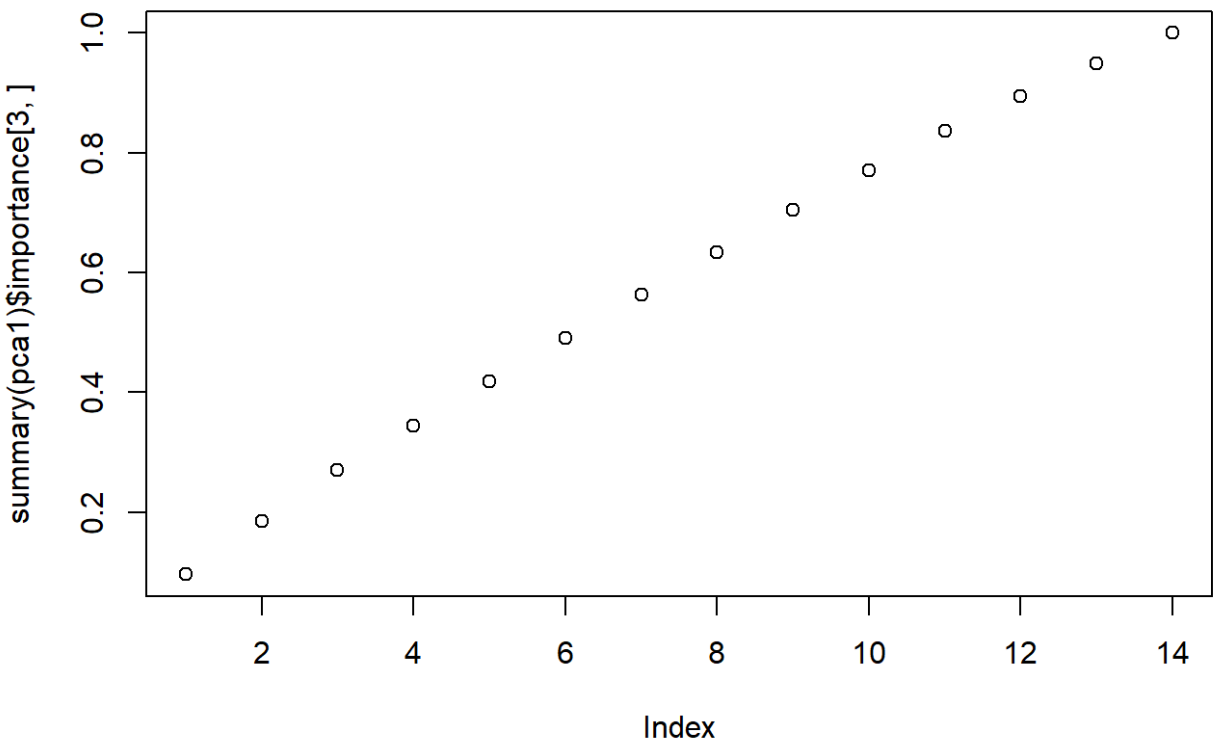
```
dataset_num <- dataset_clean
dataset_num[] <- lapply(dataset_num, as.factor)
dataset_num[] <- lapply(dataset_num, as.numeric)
dataset_num <- dataset_num[ , !(names(dataset_num) %in% c("finish_time"))]
```

Ara separem les dades entre la variable dependent (finishing_position), i les independents. Després escalem els resultats, creem el PCA i visualitzem el resultats:

```
finishing_position_y <- dataset_num$finishing_position
dataset_num$finishing_position <- NULL
```

```
finishing_position_y <- scale(finishing_position_y)
dataset_num <- scale(dataset_num)

pca1 <- prcomp(dataset_num, center=TRUE, scale.=TRUE)
plot(summary(pca1)$importance[3,])
```




```
summary(pca1)
```

```
## Importance of components:
##                             PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     1.16071 1.11797 1.08560 1.02418 1.01673 1.00905 1.00574
## Proportion of Variance 0.09623 0.08928 0.08418 0.07492 0.07384 0.07273 0.07225
## Cumulative Proportion  0.09623 0.18551 0.26969 0.34461 0.41845 0.49118 0.56343
##                             PC8     PC9    PC10    PC11    PC12   PC13    PC14
## Standard deviation     0.99863 0.98578 0.96840 0.95630 0.90073 0.8759 0.84387
## Proportion of Variance 0.07123 0.06941 0.06699 0.06532 0.05795 0.0548 0.05087
## Cumulative Proportion  0.63466 0.70407 0.77106 0.83638 0.89433 0.9491 1.00000
```

Un cop ja tenim els pca, utilitzarem aquests per a crear el model:

```
pcs <- as.data.frame(pca1$x)
dataset_pcs <- cbind(finishing_position_y, pcs)
lmodel <- lm(finishing_position_y ~ ., data = dataset_pcs)
summary(lmodel)
```

```
##
## Call:
## lm(formula = finishing_position_y ~ ., data = dataset_pcs)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -2.58189 -0.64952 -0.04013  0.62296  2.94892
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.841e-17  5.089e-03   0.000 1.000000
## PC1          2.640e-01  4.385e-03  60.210  < 2e-16 ***
## PC2          2.069e-01  4.552e-03  45.458  < 2e-16 ***
## PC3         -1.021e-01  4.688e-03 -21.769  < 2e-16 ***
## PC4         -5.482e-02  4.969e-03 -11.032  < 2e-16 ***
## PC5         -3.150e-05  5.006e-03  -0.006 0.994979
## PC6         -4.237e-02  5.044e-03  -8.401  < 2e-16 ***
## PC7          5.489e-02  5.060e-03  10.846  < 2e-16 ***
## PC8          1.699e-02  5.096e-03   3.334 0.000858 ***
## PC9         -3.278e-02  5.163e-03  -6.350 2.19e-10 ***
## PC10        -8.349e-02  5.256e-03 -15.885  < 2e-16 ***
## PC11         6.213e-02  5.322e-03  11.674  < 2e-16 ***
## PC12         2.000e-02  5.650e-03   3.539 0.000402 ***
## PC13         2.406e-02  5.810e-03   4.141 3.47e-05 ***
## PC14        -3.613e-01  6.031e-03 -59.905  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8529 on 28072 degrees of freedom
## Multiple R-squared:  0.2729, Adjusted R-squared:  0.2725
## F-statistic: 752.4 on 14 and 28072 DF,  p-value: < 2.2e-16
```

Podem veure, que el nostre model ara ha empitjorat encara més, tenim un R2 de 0.27. Així doncs, veiem que utilitzar els PCA per a millorar el model no ha donat els resultats esperats.

4. Representació dels resultats a partir de taules i gràfiques.

Anem a veure ara la reprentació dels resultats en taules i gràfics dels resultats obtinguts, ho podem fer mitjançant la següent gràfica dinàmica en la que es pot seleccionar cada un dels caballs o les curses en que estem fent l'estudi.

```
library(crosstalk)

dataset_clean$finishing_position <-  unlist(lapply(dataset_clean$finishing_position, as.facto
r))
dataset_clean$race_date <- dataset_outliers$race_date
tx <- highlight_key(dataset_clean)
widgets <- bscols(
  widths = c(12,12,12),
  filter_select("horse_name", "Horse_name", tx, ~horse_name),
  filter_checkbox("race_distance", "race_distance", tx, ~race_distance, inline = TRUE)
)
bscols(
  widths = c(4,6), widgets,
  plot_ly(tx, x = ~race_date, y = ~finish_time, showlegend = FALSE, type = 'scatter', text =
~paste("Horse name: ", horse_name, '<br>Position:', finishing_position, '<br>Jockey:', jockey
),
  color = ~race_distance, size = ~as.numeric(finishing_position)
  )
)
```

Horse_name

race_distance

☐ 1000 ☐ 1200 ☐ 1400
☐ 1600 ☐ 1650 ☐ 1800
☐ 2000 ☐ 2200 ☐ 2400