Mock 11+ standardisation R Task

```r
#Reading File
setwd("~/Desktop/Github/Mock_11+_standardisation")
JSTMT <- read.csv("Junior Statistician R Task Main Testers.csv")

#Reading dates in correct format
JSTMT$DoB <- as.Date(JSTMT$DoB)
JSTMT$DoT <- as.Date(JSTMT$DoT)


#Checking for variables' names
colnames(JSTMT)
```

```
## [1] "SchoolID" "PupilID"  "Gender"   "DoB"      "DoT"      "EAL"      "FSM"
## [8] "RawScore"
```

```r
#rows
nrow(JSTMT)
```

```
## [1] 200
```

```r
#Any missing values
sum(is.na(JSTMT))
```

```
## [1] 0
```

```r
#Converting binary/categorical variables as factors
JSTMT$SchoolID <- as.factor(JSTMT$SchoolID)
JSTMT$Gender <- as.factor(JSTMT$Gender)
JSTMT$EAL <- as.factor(JSTMT$EAL)
JSTMT$FSM <- as.factor(JSTMT$FSM)
```

```r
#Calculating age in months at day of test
#age = 12*(TestYear - BirthYear) + (TestMonth - BirthMonth) - (1 only if test 'day' before birth 'day')

JSTMT$age <- (as.numeric(format(JSTMT$DoT, "%Y")) - as.numeric(format(JSTMT$DoB, "%Y"))) * 12 +
  (as.numeric(format(JSTMT$DoT, "%m")) - as.numeric(format(JSTMT$DoB, "%m"))) -
  (as.numeric(format(JSTMT$DoT, "%d")) < as.numeric(format(JSTMT$DoB, "%d")))
```

```r
#Summary Statistics for various variables
summary(JSTMT[c("Gender", "age", "EAL", "FSM", "RawScore")])
```

```
##  Gender       age          EAL      FSM        RawScore
##  F: 91   Min.   :121.0   N:167   N:155   Min.   : 0.00
```

```
##  M:109   1st Qu.:124.0   Y: 33   Y: 45   1st Qu.:12.75
##          Median :126.0                   Median :23.00
##          Mean   :126.5                   Mean   :22.82
##          3rd Qu.:129.0                   3rd Qu.:31.00
##          Max.   :132.0                   Max.   :50.00
```

```
cat("\n Sd RawScore:\n")
```

```
##
##  Sd RawScore:
```

```
sd(JSTMT$RawScore)
```

```
## [1] 12.8918
```

```
cat("\n Sd age:\n")
```

```
##
##  Sd age:
```

```
sd(JSTMT$age)
```

```
## [1] 3.204755
```

```
cat("\n Gender prop:\n")
```

```
##
##  Gender prop:
```

```
prop.table(table(JSTMT$Gender)) * 100
```

```
##
##    F    M
## 45.5 54.5
```

```
cat("\n FSM prop:\n")
```

```
##
##  FSM prop:
```

```
prop.table(table(JSTMT$FSM)) * 100
```

```
##
##    N    Y
## 77.5 22.5
```

```r
cat("\n EAL prop:\n")
```

```
##
##  EAL prop:
```

```r
print(prop.table(table(JSTMT$EAL)) * 100)
```

```
##
##    N    Y
## 83.5 16.5
```

```r
#Summary Statistics for RawScore by school
summary(JSTMT$SchoolID)
```

```
##  1  2  3  4  5  6  8  9 10 11 12 13 14 15
## 14  9 27 10 40 14  6  9  5  3 44  7  7  5
```

```r
aggregate(RawScore ~ SchoolID, JSTMT, min)
```

```
##    SchoolID RawScore
## 1         1       12
## 2         2        8
## 3         3        0
## 4         4        8
## 5         5        0
## 6         6        0
## 7         8       13
## 8         9        0
## 9        10       10
## 10       11        0
## 11       12        4
## 12       13        0
## 13       14       11
## 14       15       22
```

```r
aggregate(RawScore ~ SchoolID, JSTMT, max)
```

```
##    SchoolID RawScore
## 1         1       49
## 2         2       34
## 3         3       31
## 4         4       48
## 5         5       49
## 6         6       43
## 7         8       49
## 8         9       44
## 9        10       42
## 10       11       24
## 11       12       50
## 12       13       33
## 13       14       33
## 14       15       30
```

```r
aggregate(RawScore ~ SchoolID, JSTMT, mean)
```

```
##    SchoolID RawScore
## 1         1 29.64286
## 2         2 19.66667
## 3         3 14.03704
## 4         4 28.90000
## 5         5 20.05000
## 6         6 19.50000
## 7         8 30.66667
## 8         9 21.00000
## 9        10 21.40000
## 10       11  8.00000
## 11       12 28.63636
## 12       13 22.28571
## 13       14 24.00000
## 14       15 28.20000
```

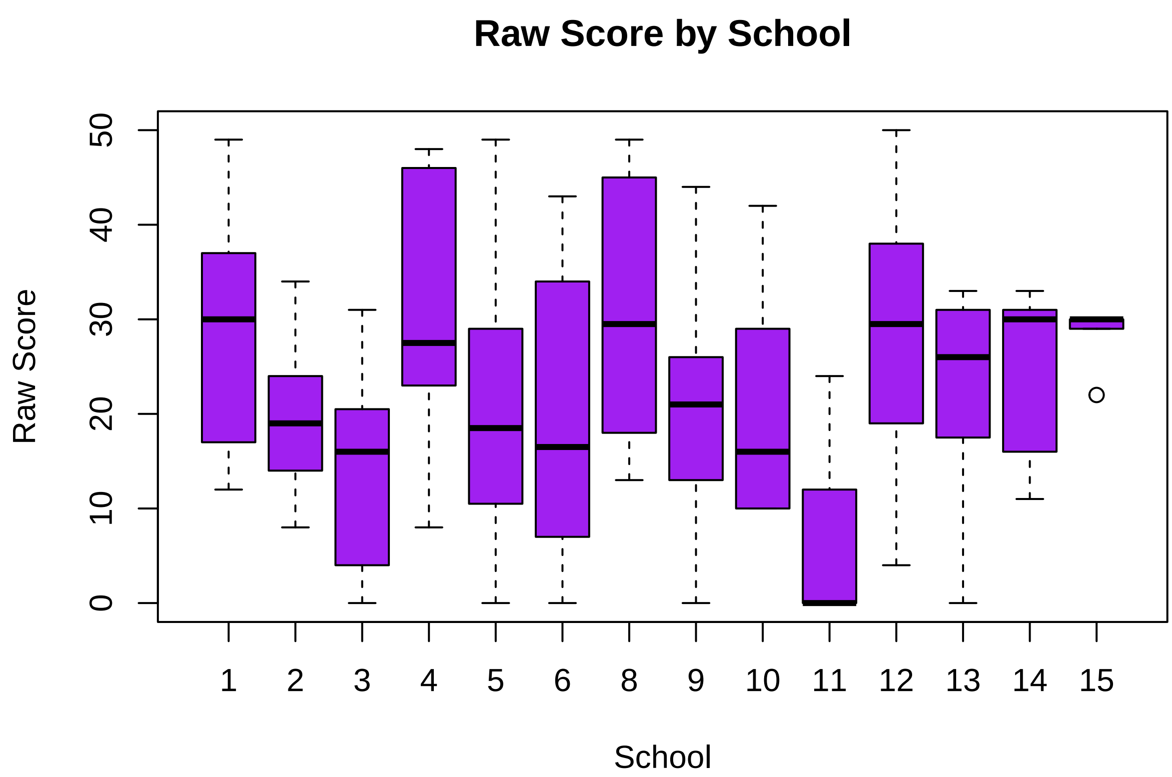
```r
aggregate(RawScore ~ SchoolID, JSTMT, median)
```

```
##    SchoolID RawScore
## 1         1     30.0
## 2         2     19.0
## 3         3     16.0
## 4         4     27.5
## 5         5     18.5
## 6         6     16.5
## 7         8     29.5
## 8         9     21.0
## 9        10     16.0
## 10       11      0.0
## 11       12     29.5
## 12       13     26.0
## 13       14     30.0
## 14       15     30.0
```

```r
aggregate(RawScore ~ SchoolID, JSTMT, sd)
```

```
##    SchoolID  RawScore
## 1         1 10.695845
## 2         2  9.578622
## 3         3  9.924645
## 4         4 14.348442
## 5         5 11.734127
## 6         6 14.015102
## 7         8 14.264174
## 8         9 12.980755
## 9        10 13.885244
## 10       11 13.856406
## 11       12 12.875235
## 12       13 12.486183
## 13       14  9.255629
## 14       15  3.492850
```

```r
boxplot(RawScore ~ SchoolID,
        data = JSTMT,
        col = "purple",
        xlab = "School",
        ylab = "Raw Score",
        main = "Raw Score by School")
```




```r
#Summary Statistics for RawScore by gender

aggregate(RawScore ~ Gender, JSTMT, min)
```

```
##   Gender RawScore
## 1      F        0
## 2      M        0
```

```r
aggregate(RawScore ~ Gender, JSTMT, max)
```

```
##   Gender RawScore
## 1      F       49
## 2      M       50
```

```r
aggregate(RawScore ~ Gender, JSTMT, mean)
```

```
##   Gender RawScore
## 1      F 22.49451
## 2      M 23.09174
```

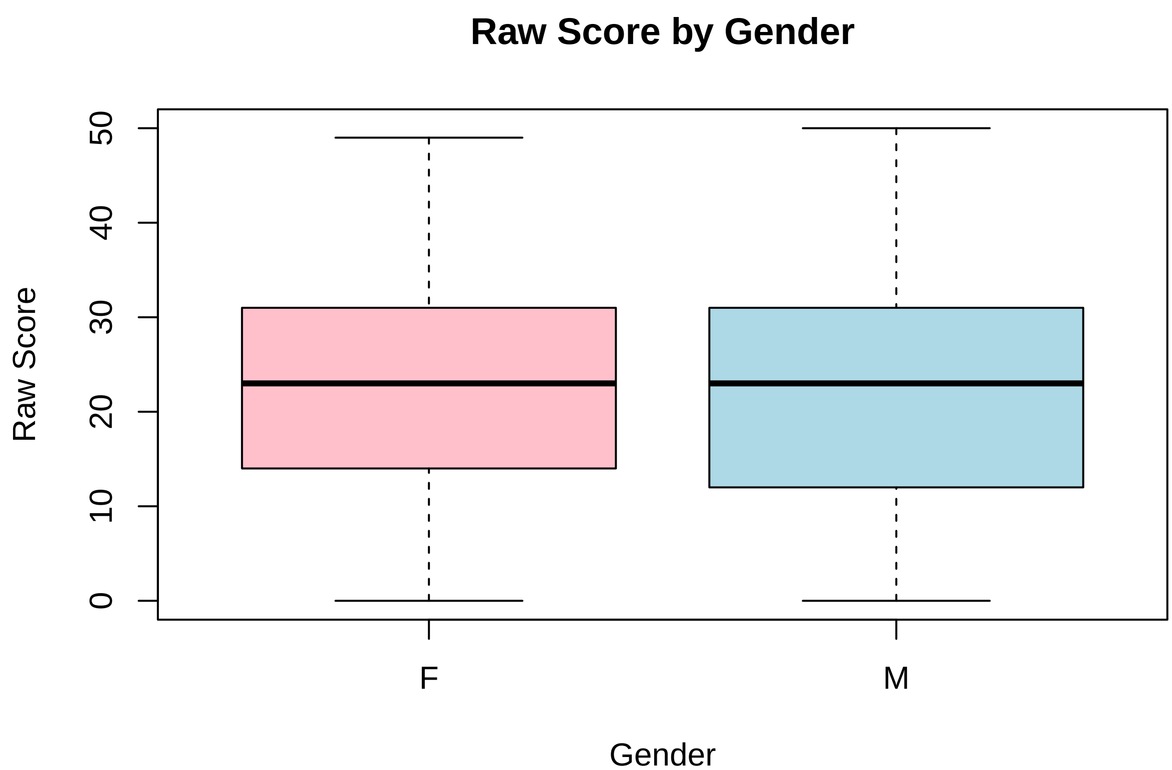
```r
aggregate(RawScore ~ Gender, JSTMT, median)
```

```
##   Gender RawScore
## 1      F       23
## 2      M       23
```

```r
aggregate(RawScore ~ Gender, JSTMT, sd)
```

```
##   Gender RawScore
## 1      F 12.18412
## 2      M 13.50414
```

```r
boxplot(RawScore ~ Gender,
        data = JSTMT,
        col = c("pink", "lightblue"),
        xlab = "Gender",
        ylab = "Raw Score",
        main = "Raw Score by Gender")
```



```r
#Summary Statistics for RawScore by age

table(JSTMT$age)
```

```
##
## 121 122 123 124 125 126 127 128 129 130 131 132
##  12  15  15  20  21  19  20  15  19  19  13  12
```

```r
aggregate(RawScore ~ age, JSTMT, min)
```

```
##    age RawScore
## 1  121        0
## 2  122        0
## 3  123        0
## 4  124        0
## 5  125        0
## 6  126        7
## 7  127        5
## 8  128        7
## 9  129        8
## 10 130       12
## 11 131       16
## 12 132       29
```

```r
aggregate(RawScore ~ age, JSTMT, max)
```

```
##    age RawScore
## 1  121       29
## 2  122       23
## 3  123       34
## 4  124       35
## 5  125       42
## 6  126       37
## 7  127       48
## 8  128       38
## 9  129       48
## 10 130       50
## 11 131       50
## 12 132       50
```

```r
aggregate(RawScore ~ age, JSTMT, mean)
```

```
##    age  RawScore
## 1  121 11.416667
## 2  122  9.466667
## 3  123 15.266667
## 4  124 17.500000
## 5  125 20.000000
## 6  126 20.894737
## 7  127 23.400000
## 8  128 27.000000
```

```
## 9  129 28.105263
## 10 130 27.736842
## 11 131 36.000000
## 12 132 40.583333
```

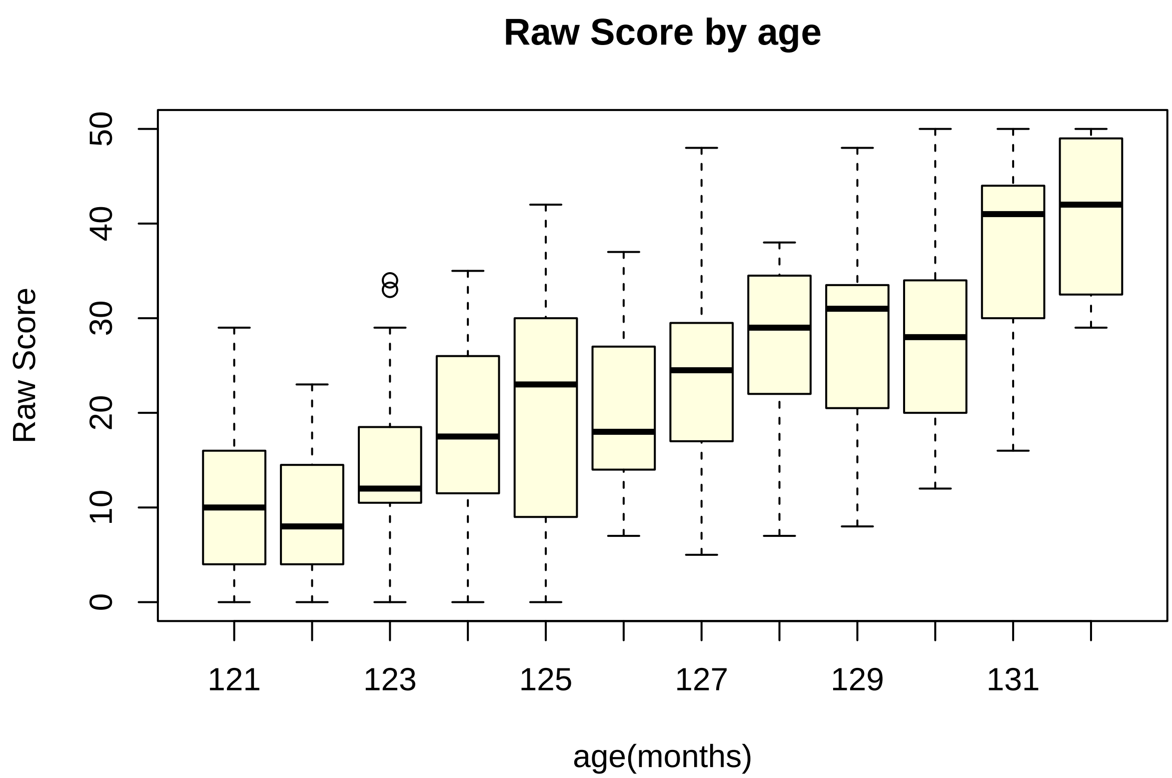
```r
aggregate(RawScore ~ age, JSTMT, median)
```

```
##    age RawScore
## 1  121     10.0
## 2  122      8.0
## 3  123     12.0
## 4  124     17.5
## 5  125     23.0
## 6  126     18.0
## 7  127     24.5
## 8  128     29.0
## 9  129     31.0
## 10 130     28.0
## 11 131     41.0
## 12 132     42.0
```

```r
aggregate(RawScore ~ age, JSTMT, sd)
```

```
##    age  RawScore
## 1  121  9.894519
## 2  122  7.763161
## 3  123 10.215301
## 4  124 10.308402
## 5  125 12.593649
## 6  126  9.188729
## 7  127 10.404452
## 8  128  8.976159
## 9  129 10.826381
## 10 130 10.712620
## 11 131 11.518102
## 12 132  8.691253
```

```r
boxplot(RawScore ~ age,
        data = JSTMT,
        col = "lightyellow",
        xlab = "age(months)",
        ylab = "Raw Score",
        main = "Raw Score by age")
```




```r
#Summary Statistics for RawScore by EAL

aggregate(RawScore ~ EAL, JSTMT, min)
```

```
##   EAL RawScore
## 1   N        0
## 2   Y        2
```

```r
aggregate(RawScore ~ EAL, JSTMT, max)
```

```
##   EAL RawScore
## 1   N       50
## 2   Y       49
```

```r
aggregate(RawScore ~ EAL, JSTMT, mean)
```

```
##   EAL RawScore
## 1   N 22.62874
## 2   Y 23.78788
```

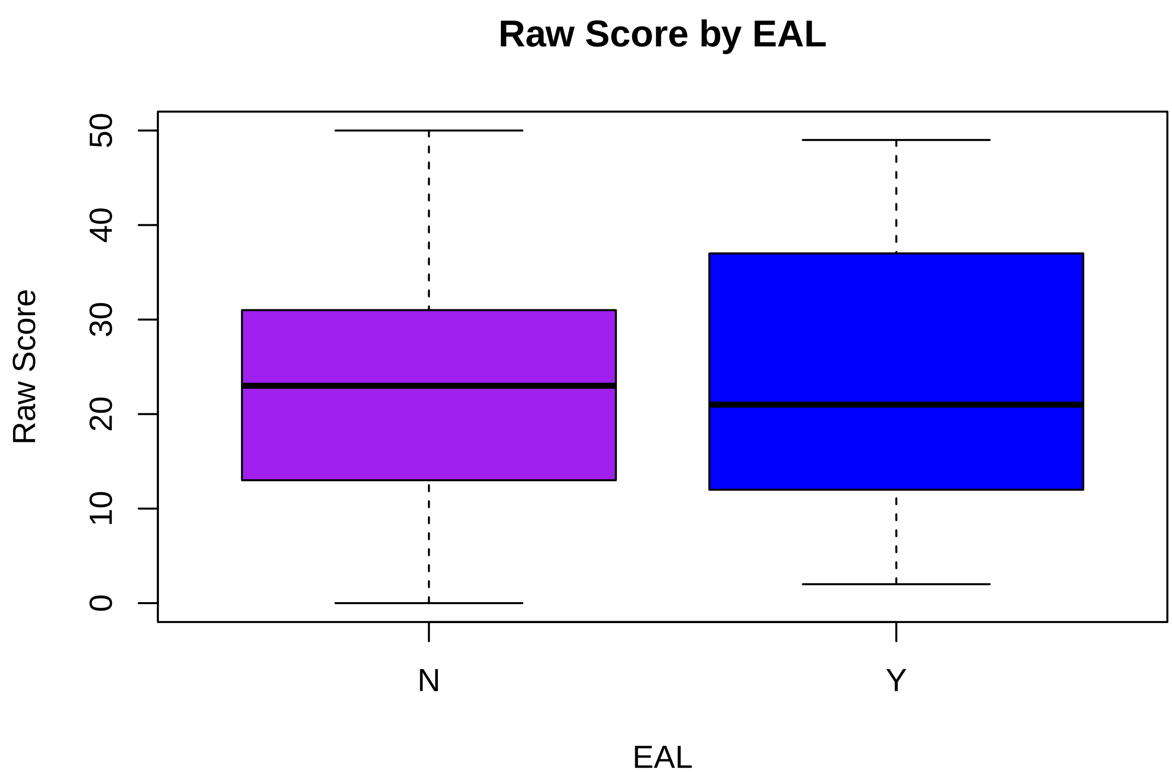
```r
aggregate(RawScore ~ EAL, JSTMT, median)
```

```
##   EAL RawScore
## 1   N       23
## 2   Y       21
```

```r
aggregate(RawScore ~ EAL, JSTMT, sd)
```

```
##   EAL RawScore
## 1   N 12.74329
## 2   Y 13.78350
```

```r
boxplot(RawScore ~ EAL,
        data = JSTMT,
        col = c("purple", "blue"),
        xlab = "EAL",
        ylab = "Raw Score",
        main = "Raw Score by EAL")
```




```r
#Summary Statistics for RawScore by FSM

aggregate(RawScore ~ FSM, JSTMT, min)
```

```
##   FSM RawScore
## 1   N        0
## 2   Y        0
```

```r
aggregate(RawScore ~ FSM, JSTMT, max)
```

```
##   FSM RawScore
## 1   N       50
## 2   Y       39
```

```r
aggregate(RawScore ~ FSM, JSTMT, mean)
```

```
##   FSM RawScore
## 1   N 23.99355
## 2   Y 18.77778
```

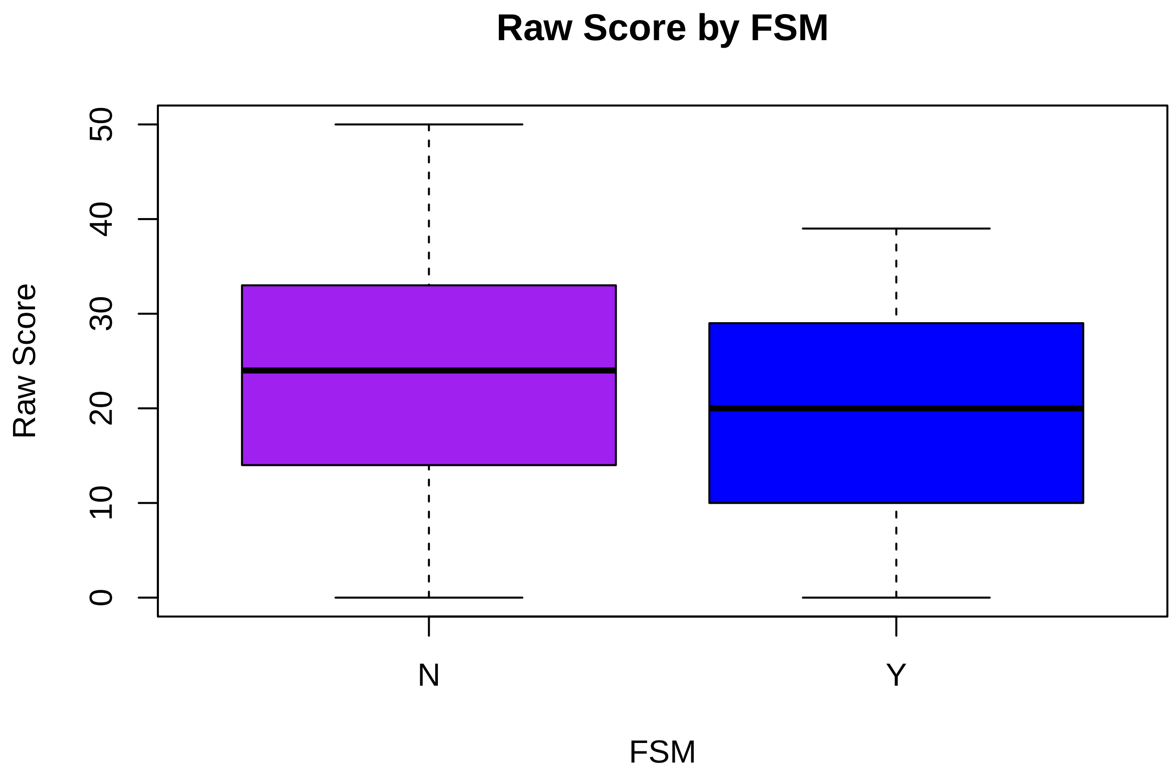
```r
aggregate(RawScore ~ FSM, JSTMT, median)
```

```
##   FSM RawScore
## 1   N       24
## 2   Y       20
```

```r
aggregate(RawScore ~ FSM, JSTMT, sd)
```

```
##   FSM RawScore
## 1   N 13.10670
## 2   Y 11.35159
```

```r
boxplot(RawScore ~ FSM,
        data = JSTMT,
        col = c("purple", "blue"),
        xlab = "FSM",
        ylab = "Raw Score",
        main = "Raw Score by FSM")
```



(1)

`Raw Score v School`

Clear association between Raw Score and School

- School 11 shows the worst performance (mean = 8, median = 0, SD = 13.9). However, the sample size is very small (n = 3), so it is difficult to generalise from this result.
- School 3 shows poor performance, with the second-lowest results (mean = 14, median = 16, SD = 9.9). This indicates that most pupils are achieving low scores, and the lower variability suggests these weaker results are clustered rather than spread out.
- The highest score achieved in this school is 31/50, while the lowest is 0, further reinforcing that overall performance is considerably below that of other schools.
- With a sample size of 27 pupils, these results are more generalisable, indicating that this school's test performance is relatively poor.
- School 12 has the best performance (min = 4, max = 50, mean = 28.6, median = 29.5, SD = 12.9). As the sample size is 44, I am more confident in this initial conclusion.
- School 15 appears to have excellent performance (mean = 28.2, median = 30, SD = 3.5), but the sample size is low, n = 5. Although the results look strong, it is uncertain whether they truly reflect the school population as a whole.

`Raw Score v Gender`

- No meaningful association (both genders show very similar results) - but male scores are more variable as seen in the boxplot, males Raw Scores have a SD of 13.5 and compared to 12.2 for females. This greater variability may reflect a wider range of behavioural, engagement, or learning patterns seen within the male group.

Raw Score v Age

- In general, Mean and Median results across ages show that Raw Score increases as pupils get older. This is expected, as age is strongly linked to cognitive development, skill growth, test-taking maturity, and general learning exposure.
- Younger ages have some minimum scores of 0 (no real attempt, exceptional/personal circumstances, behaviour issues).
- Although from age 125 months to 126 months there is dip in median and again from 129 months to 130 months, the overall pattern still shows Raw Score increases as pupils get older.

Raw Score v EAL

- Mean and median raw scores are slightly lower, but not by much, for pupils who have English as an additional language compared to those who do not. This would be expected, possibly due to difficulties in understanding the vocabulary in certain questions, but the difference is quite small.
- This difference could also be small because, in the UK, many pupils whose additional language is English come from Commonwealth countries where English is already a second language, which may reduce the gap.
- There is slightly higher variability in scores for pupils with English as an additional language (larger IQR range and SD: 13.8 vs 12.7), possibly due to a wider range of demographics/ethnicities and different levels of prior exposure to English.
- The boxplot shows that pupils whose additional language is English have a noticeably higher upper-quartile score (75th percentile). One potential reason could be that, in some South Asian communities, pressure and attitudes towards education are particularly strong, with higher expectations and a greater focus on tuition and extra studies as part of culture.

Raw Score v FSM

- There is a significant difference in raw score means and medians. Pupils receiving free school meals (FSM) have a mean raw score of 18.8 compared with 24 for non FSM pupils. The median is 20 for FSM and 24 for non FSM. The maximum raw score obtained by an FSM pupil is 39, compared with 50 for non-FSM pupils. There is also higher variability in the non FSM group (SD = 13.1) compared with the FSM group (SD = 11.4), suggesting less variation in scores within the FSM group (FSM pupils maybe more similar to each other).This association is clearly shown in the boxplots.
- Pupils eligible for free school meals are more likely to come from lower socio-economic backgrounds, which can influence educational outcomes.
- Factors such as lower household income, ethnicity, parental or family circumstances, reduced access to learning resources (books, iPads, TV, computers, tuition), disabilities, and differences in home educational support may contribute to the lower scores seen in the FSM group.

Important Point

- Although there are clear associations in the descriptive statistics, formal statistical tests would be required to determine whether these differences are statistically significant and can be generalised to the wider population.

(2)

- Raw scores are adjusted by using a statistical model (ordinal logistic model) to estimate how much of a pupil's raw score is explained by confounding variables rather than ability.
- The model gives the cumulative probability that a pupil with those characteristics would score at or below their observed raw score.
- This cumulative probability represents the pupil's percentile position after adjusting for the confounders.
- That percentile is then mapped onto the N(100, 15) scale to produce a standardised score that reflects ability only, with the influence of the confounding variables removed.

```r
#Ordinal Logistic Regression Model
JSTMT$RawScore <- factor(JSTMT$RawScore,
                         levels = 0:50,
                         ordered = TRUE)

library(MASS)
olm <- polr(RawScore ~ age, data = JSTMT, method = "logistic", Hess = TRUE)
summary(olm)
```

```
## Call:
## polr(formula = RawScore ~ age, data = JSTMT, Hess = TRUE, method = "logistic")
##
## Coefficients:
##      Value Std. Error t value
## age 0.4118    0.04644   8.868
##
## Intercepts:
##       Value   Std. Error t value
## 0|1   48.3560  5.7786     8.3681
## 1|2   48.4748  5.7788     8.3884
## 2|3   48.5826  5.7790     8.4068
## 3|4   48.5826  5.7790     8.4068
## 4|5   49.1035  5.7840     8.4896
## 5|6   49.3102  5.7874     8.5203
## 6|7   49.4935  5.7909     8.5468
## 7|8   49.6569  5.7940     8.5703
## 8|9   49.8978  5.7991     8.6045
## 9|10  50.0284  5.8020     8.6226
## 10|11 50.2662  5.8076     8.6553
## 11|12 50.4490  5.8124     8.6796
## 12|13 50.6229  5.8177     8.7016
## 13|14 50.7599  5.8227     8.7176
## 14|15 50.8921  5.8278     8.7327
## 15|16 50.9869  5.8313     8.7437
## 16|17 51.1106  5.8361     8.7577
## 17|18 51.3789  5.8470     8.7872
```

```
## 18|19 51.5474  5.8535     8.8063
## 19|20 51.6016  5.8554     8.8126
## 20|21 51.9174  5.8667     8.8495
## 21|22 51.9949  5.8694     8.8586
## 22|23 52.0206  5.8703     8.8617
## 23|24 52.1494  5.8748     8.8768
## 24|25 52.3795  5.8825     8.9043
## 25|26 52.4822  5.8858     8.9167
## 26|27 52.5352  5.8877     8.9229
## 27|28 52.6167  5.8908     8.9320
## 28|29 52.6714  5.8928     8.9382
## 29|30 52.9818  5.9042     8.9736
## 30|31 53.2912  5.9160     9.0080
## 31|32 53.5252  5.9246     9.0344
## 32|33 53.5597  5.9258     9.0383
## 33|34 53.7385  5.9319     9.0592
## 34|35 54.0489  5.9412     9.0973
## 35|36 54.2679  5.9472     9.1249
## 36|37 54.3650  5.9501     9.1368
## 37|38 54.4702  5.9536     9.1492
## 38|39 54.6413  5.9593     9.1692
## 39|40 54.7017  5.9612     9.1764
## 40|41 54.8296  5.9651     9.1917
## 41|42 55.0434  5.9718     9.2172
## 42|43 55.2054  5.9769     9.2364
## 43|44 55.2936  5.9797     9.2469
## 44|45 55.3867  5.9825     9.2581
## 45|46 55.4854  5.9853     9.2702
## 46|47 55.7028  5.9909     9.2979
## 47|48 55.8234  5.9935     9.3140
## 48|49 56.1078  5.9999     9.3515
## 49|50 56.7244  6.0158     9.4293
##
## Residual Deviance: 1385.693
## AIC: 1487.693
```

```r
exp(coef(olm))
```

```
##      age
## 1.509576
```

```r
2 * (1 - pnorm(abs(summary(olm)$coefficients["age", "t value"])))
```

```
## [1] 0
```

(3)

Raw Scores are treated as an ordinal outcome.

Zeta coefficient

- Zeta is the intercept for the `lth` cutoff. It represents the log odds of scoring at or below Raw Score `l` when the predictor (age) is equal to 0. If you apply the inverse logit function to Zeta, you get the baseline cumulative probability of scoring at or below that level. However, this baseline probability is not very meaningful in practice because age $= 0$ is outside the realistic range for pupil ages.

Age statistical significance

- Age is a highly significant predictor variable, as shown by the large t-value and p value 0.

Eta coefficient

- The coefficient (eta = 0.4118) means that for every one month increase in age, the log odds of being in a lower raw score category decrease by 0.4118.

Odds Ratio

- For every one month increase in age, the odds of achieving a higher score increase by about 51% (OR $e\hat{}0.4118 = 1.51$).
- For every one month increase in age, the odds of getting a lower score decrease by about 34% (OR $e\hat{}-0.4118 = 0.66$).

Simple Examples

- If a pupil currently has four chances out of ten of getting a high score, a pupil who is just one month older would typically have around six chances out of ten.
- If a pupil has six chances out of ten of ending up in a low score category, another pupil who is only one month older would see that chance drop to roughly four out of ten.

```r
library(generalhoslem)
```

```
## Loading required package: reshape
```

```r
lipsitz.test(olm, g = 10)
```

```
## Warning in lipsitz.test(olm, g = 10): g >= n/5c. Running this test when g >=
## n/5c is not recommended.

##
##  Lipsitz goodness of fit test for ordinal response models
##
## data:  formula:  RawScore ~ age
## LR statistic = 6.1903, df = 9, p-value = 0.7207
```

(4)

- The Lipsitz test checks for a general type of misfit in an ordinal logistic model. It looks to see whether the predicted probabilities for each score category match the observed data across different groups of pupils. If the model is systematically predicting the chances of low or high scores too high or too low in any part of the range, the test will pick this up. So the form of misfit it assesses is whether the model's predictions are poorly calibrated across the score distribution.
- In this case, the p value was not significant. This means we do not reject the null hypothesis that the model fits correctly. There is no evidence that the model is systematically over or under predicting raw score probabilities, and therefore the ordinal logistic model appears to fit the data adequately.

(5)

```r
getAnywhere("lipsitz.test")
```

```
## A single object matching 'lipsitz.test' was found
## It was found in the following places
##   package:generalhoslem
##   namespace:generalhoslem
## with value
##
## function (model, g = 10)
## {
##     oldmodel <- model
##     if (class(oldmodel) == "polr") {
##         yhat <- as.data.frame(fitted(oldmodel))
##     }
##     else if (class(oldmodel) == "clm") {
##         predprob <- oldmodel$model[, 2:ncol(oldmodel$model),
##             drop = F]
##         yhat <- as.data.frame(predict(oldmodel, newdata = predprob,
##             type = "prob")$fit)
##     }
##     else warning("Model is not of class polr or clm. Test may fail.")
##     formula <- formula(oldmodel$terms)
##     DNAME <- paste("formula: ", deparse(formula))
##     METHOD <- "Lipsitz goodness of fit test for ordinal response models"
##     obs <- oldmodel$model[1]
##     if (g < 6)
##         warning("g < 6. Running this test when g < 6 is not recommended.")
##     if (g >= nrow(obs)/(5 * ncol(yhat)))
##         warning("g >= n/5c. Running this test when g >= n/5c is not recommended.")
##     yhat$score <- apply(sapply(1:ncol(yhat), function(i) {
##         yhat[, i] * i
##     }), 1, sum)
##     yhat$tmp <- 1:nrow(yhat)
##     yhat <- yhat[order(yhat$score), ]
##     cutyhats <- cut(1:nrow(yhat), breaks = g, include.lowest = T)
##     cutyhats <- cutyhats[order(yhat$tmp)]
##     yhat <- yhat[order(yhat$tmp), ]
##     yhat$score <- NULL
##     yhat$tmp <- NULL
##     dfobs <- data.frame(obs, cutyhats)
##     dfobsmelt <- melt(dfobs, id.vars = 2)
##     observed <- cast(dfobsmelt, cutyhats ~ value, length)
##     if (g != nrow(observed)) {
##         warning(paste("Not possible to compute", g, "rows. There might be too few observations."))
##     }
##     oldmodel$model <- cbind(oldmodel$model, cutyhats = dfobs$cutyhats)
##     oldmodel$model$grp <- as.factor(vapply(oldmodel$model$cutyhats,
##         function(x) which(observed[, 1] == x), 1))
##     newmodel <- update(oldmodel, . ~ . + grp, data = oldmodel$model)
##     if (class(oldmodel) == "polr") {
##         LRstat <- oldmodel$deviance - newmodel$deviance
##     }
##     else if (class(oldmodel) == "clm") {
```

```
##         LRstat <- abs(-2 * (newmodel$logLik - oldmodel$logLik))
##     }
##     PARAMETER <- g - 1
##     PVAL <- 1 - pchisq(LRstat, PARAMETER)
##     names(LRstat) <- "LR statistic"
##     names(PARAMETER) <- "df"
##     structure(list(statistic = LRstat, parameter = PARAMETER,
##         p.value = PVAL, method = METHOD, data.name = DNAME, newmoddata = oldmodel$model,
##         predictedprobs = yhat), class = "htest")
## }
## <bytecode: 0x125b263b8>
## <environment: namespace:generalhoslem>
```

(6)

`c` is the number of outcome categories (Ordinal Raw Scores)

```r
library(brant)
brant(olm)
```

```
## --------------------------------------------
## Test for X2  df  probability
## --------------------------------------------
## Omnibus      50.09   49  0.43
## age      50.09   49  0.43
## --------------------------------------------
##
## H0: Parallel Regression Assumption holds
```

(7)

- The proportional odds assumption means that the effect of the predictor (age) is the same across all splits of the ordinal outcome (RawScore).
- A 1 month increase has the same effect whether you move from score 1 to 2, 2 to 3 and so on.
- Brant Test - probabilities higher than 5% significance level, failed to reject H0, no evidence that Parallel Regression Assumption (proportional odds assumption) doesn't hold.

```r
#Predicted probabilities for each raw score category (0, 1, ..., 50) for each pupil based on their age.
#For each pupil, this gives P(RawScore = k | age) for k = 0,...,50.

predicted_raw_prob <- predict(olm, newdata = JSTMT, type = "probs")

#Cumulative probabilities for each pupil, for each score k, this gives P(RawScore   k | age)
predicted_cumprob <- t(apply(predicted_raw_prob, 1, cumsum))


JSTMT$RawScore <- as.character(JSTMT$RawScore)
JSTMT$RawScore <- as.numeric(JSTMT$RawScore)



#Extracting the cumulative probability for each pupil's actual score
```

```r
cumprob_obs_score <- predicted_cumprob[cbind(1:nrow(JSTMT), JSTMT$RawScore + 1)]

#Converting percentiles to SAS
SAS <- qnorm(cumprob_obs_score, mean = 100, sd = 15)

SAS[SAS < 69]  <- 69
SAS[SAS > 141] <- 141
SAS <- round(SAS)

JSTMT$CumProb <- cumprob_obs_score
JSTMT$SAS <- SAS
```

```r
#Summary Statistics SAS (overall)
summary(JSTMT$SAS)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    73.0    90.0   101.0   101.3   112.0   141.0
```

```r
sd(JSTMT$SAS)
```

```
## [1] 14.84598
```

```r
qnorm(0.25, mean = 100, sd = 15)
```

```
## [1] 89.88265
```

```r
qnorm(0.75, mean = 100, sd = 15)
```

```
## [1] 110.1173
```

```r
#Summary Statistics SAS (by age)

table(JSTMT$age)
```

```
##
## 121 122 123 124 125 126 127 128 129 130 131 132
##  12  15  15  20  21  19  20  15  19  19  13  12
```

```r
aggregate(SAS ~ age, JSTMT, min)
```

```
##    age SAS
## 1  121  87
## 2  122  83
## 3  123  80
## 4  124  77
## 5  125  74
## 6  126  80
## 7  127  75
## 8  128  75
```

```
## 9   129  73
## 10  130  76
## 11  131  76
## 12  132  87
```

```r
aggregate(SAS ~ age, JSTMT, max)
```

```
##    age SAS
## 1  121 126
## 2  122 117
## 3  123 128
## 4  124 126
## 5  125 130
## 6  126 122
## 7  127 130
## 8  128 117
## 9  129 125
## 10 130 141
## 11 131 141
## 12 132 141
```

```r
aggregate(SAS ~ age, JSTMT, mean)
```

```
##    age       SAS
## 1  121 104.25000
## 2  122  98.40000
## 3  123 102.93333
## 4  124 102.40000
## 5  125 102.00000
## 6  126 100.36842
## 7  127  99.50000
## 8  128 101.66667
## 9  129  99.57895
## 10 130  96.05263
## 11 131 104.15385
## 12 132 109.33333
```

```r
aggregate(SAS ~ age, JSTMT, median)
```

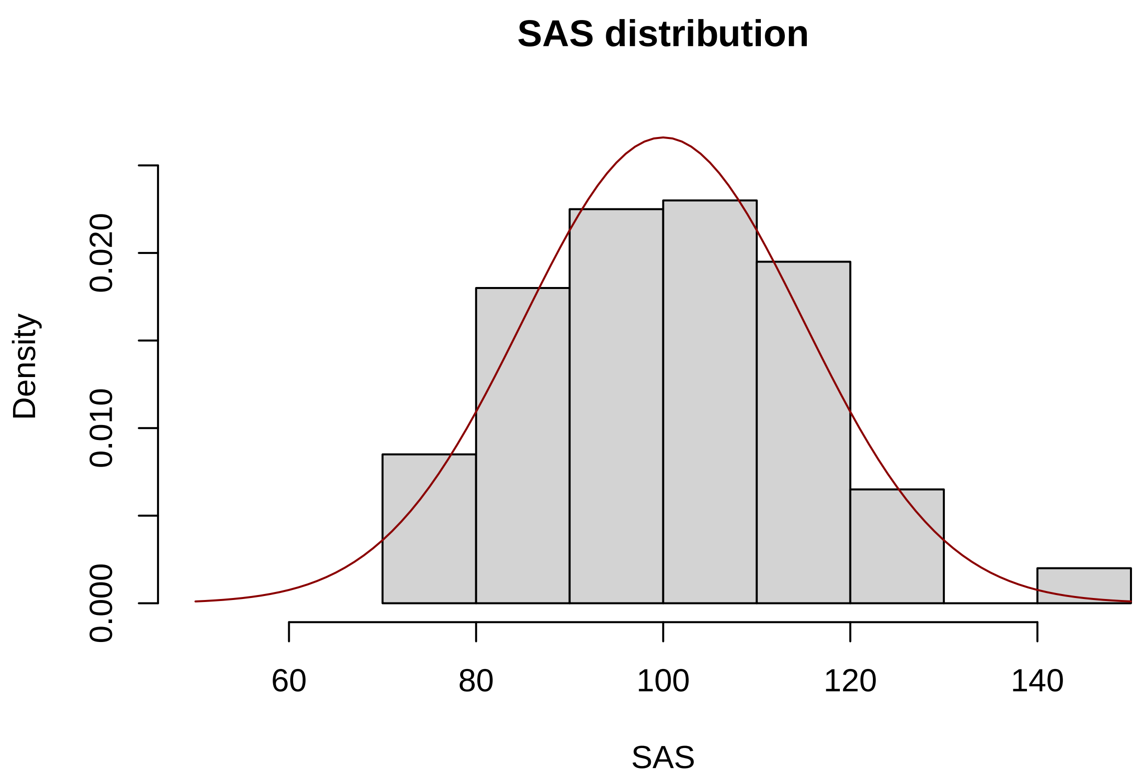
```
##    age   SAS
## 1  121 104.0
## 2  122  97.0
## 3  123 100.0
## 4  124 102.5
## 5  125 106.0
## 6  126  97.0
## 7  127 101.5
## 8  128 103.0
## 9  129 104.0
## 10 130  92.0
## 11 131 110.0
## 12 132 107.5
```

```r
aggregate(SAS ~ age, JSTMT, sd)
```

```
##    age      SAS
## 1  121 13.57889
## 2  122 11.99286
## 3  123 14.20496
## 4  124 14.28433
## 5  125 17.18139
## 6  126 12.65443
## 7  127 13.80122
## 8  128 12.49381
## 9  129 14.58049
## 10 130 16.85386
## 11 131 17.99929
## 12 132 18.70991
```

```r
#Overall SAS distribution
hist(SAS, freq = F,
     main = "SAS distribution",
     xlab = "SAS",
     xlim = c(50, 150),
     ylim = c(0, 0.027))

curve(dnorm(x, mean = 100, sd = 15),
      col = "darkred", lwd = 1, add = TRUE)
```



(8)

SAS mean = 101.3 SD = 14.8

- Natural Sampling Variation meaning the sample is only one realisation of the population.
- Due to the fact raw scores are discrete and the final SAS values are rounded to whole numbers, the resulting SAS distribution cannot be perfectly smooth and therefore cannot match an exact mean of 100 and standard deviation of 15.
- Truncation - capping SAS below 69 and above 141 removes extreme values, reducing the standard deviation and shifting the mean slightly. Also, different truncation limits would change how many SAS values are capped, which in turn affects the overall mean and standard deviation.

(9)

- The histogram and descriptive statistics for SAS indicate only a very small deviation from the target Normal(100,15) distribution. The SAS values show a tiny right skew and a slightly flatter shape, with a mildly heavier right tail (reflects truncation at 141). The interquartile range and quantiles are very close to the theoretical values (the observed lower quartile is 90, theoretical 89.9 and the observed upper quartile is 112, theoretical 110.1).

(10)

- The only concern is that the SAS distribution is not perfectly Normal. It shows a tiny right skew, but still symmetric, slightly flatter peak, and a slightly heavier upper tail, meaning the standardisation does not match fully the ideal Normal(100,15) curve exactly.

(11)

- The SAS distribution is still a very good approximation to Normal(100,15), and small imperfections are expected because scores are rounded to whole numbers and truncated at 69 and 141. The gap between 130 and 140 in the histogram reflects sampling variation as there happen to be no pupils whose cumulative probabilities mapped into that range and this does not indicate any issue with the standardisation process.

```
JSTLT <- read.csv("Junior Statistician R Task Late Testers.csv")

#Reading dates in correct format
JSTLT$DoB <- as.Date(JSTLT$DoB)
JSTLT$DoT <- as.Date(JSTLT$DoT)
```

```
#Duplicates
duplicates <- intersect(JSTMT$PupilID, JSTLT$PupilID)
duplicates
```

```
## [1]  9 61 67 82 91
```

```
length(duplicates)
```

```
## [1] 5
```

```r
JSTLT <- JSTLT[!(JSTLT$PupilID %in% JSTMT$PupilID), ]

rownames(JSTLT) <- NULL
```

(12)

5 pupils

(13)

- If a pupil appears in both files, the late test entry is a mistake. Keeping it would count the same pupil twice and produce inconsistent and inaccurate summary statistics/graphs for different groups and SAS distribution for the late testers dataset.

```r
#Checking if age was provided correctly in the late testers dataset
JSTLT$age_calculated <- (as.numeric(format(JSTLT$DoT, "%Y")) - as.numeric(format(JSTLT$DoB, "%Y"))) * 1
  (as.numeric(format(JSTLT$DoT, "%m")) - as.numeric(format(JSTLT$DoB, "%m"))) -
  (as.numeric(format(JSTLT$DoT, "%d")) < as.numeric(format(JSTLT$DoB, "%d")))

#age has been incorrectly provided in the late testers dataset
```

```r
range(JSTLT$SAS)
```

```
## [1]  69 141
```

```r
#Checking if my Ordinal Logistic Model is correct by fitting it onto Age provided in the testers datase


#Name correction
names(JSTLT)[names(JSTLT) == "Age"] <- "age"

#Computing SAS values for the Late Testers dataset on age provided (wrong age)

JSTLT$RawScore <- factor(JSTLT$RawScore,
                         levels = 0:50,
                         ordered = TRUE)

predicted_raw_prob_LT <- predict(olm, newdata = JSTLT, type = "probs")

predicted_cumprob_LT <- t(apply(predicted_raw_prob_LT, 1, cumsum))

raw_numeric_LT <- as.numeric(as.character(JSTLT$RawScore))

cumprob_obs_score_LT <- predicted_cumprob_LT[cbind(1:nrow(JSTLT), raw_numeric_LT + 1)]

SAS_LT_Computed <- qnorm(cumprob_obs_score_LT, mean = 100, sd = 15)

SAS_LT_Computed[SAS_LT_Computed < 69]  <- 69
SAS_LT_Computed[SAS_LT_Computed > 141] <- 141
SAS_LT_Computed <- round(SAS_LT_Computed)

JSTLT$SAS_LT_Computed_Wrong_Age <- SAS_LT_Computed

#SAS values provided correspond to the SAS values calculated based on age provided (wrong age) - model
```

```r
#age provided is removed from late testers dataset and replaced with age calculated
JSTLT <- subset(JSTLT, select = -c(age, SAS, SAS_LT_Computed_Wrong_Age))
names(JSTLT)[names(JSTLT) == "age_calculated"] <- "age"
```

```r
#Summary Statistics Late Testers Dataset
JSTLT$RawScore <- as.numeric(as.character(JSTLT$RawScore))
JSTLT$SchoolID <- as.factor(JSTLT$SchoolID)
JSTLT$Gender <- as.factor(JSTLT$Gender)
JSTLT$EAL <- as.factor(JSTLT$EAL)
JSTLT$FSM <- as.factor(JSTLT$FSM)
summary(JSTLT[c("Gender", "age", "EAL", "FSM", "RawScore")])
```

```
##  Gender      age          EAL     FSM        RawScore    
##  F:49   Min.   :121.0   N:73   N:73   Min.   : 0.00  
##  M:46   1st Qu.:124.0   Y:22   Y:22   1st Qu.:17.00  
##         Median :127.0                 Median :26.00  
##         Mean   :127.3                 Mean   :25.12  
##         3rd Qu.:131.0                 3rd Qu.:33.00  
##         Max.   :133.0                 Max.   :50.00  
```

```r
#Checks
table(JSTLT$SchoolID)
```

```
## 
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 
##  6  3  8  3 23  8  3  2  2  1  1 22  7  6 
```

```r
table(JSTLT$age)
```

```
## 
## 121 122 123 124 125 126 127 128 129 130 131 132 133 
##   5   6   8   6   8  10   8   5   4   6  13  10   6 
```

```r
colSums(is.na(JSTLT))
```

```
## SchoolID  PupilID   Gender      DoB      DoT      EAL      FSM RawScore 
##        0        0        0        0        0        0        0        0 
##      age 
##        0 
```

```r
duplicated(JSTLT)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

(14)

Checking for mistakes in Late Testers Dataset:

- Column Names (Age, A capital, age computation might be incorrect)
- Range of SAS provided
- Range of Raw Scores
- Any unusual levels/inconsistent names for binary/categorical variables
- Missing Values
- Duplicated Rows

Differences:

- Max age is 132 months in Main dataset, 133 in Late Dataset

- In the Late Testers dataset, pupils appear from School 7 but not School 15, whereas in the Main Testers dataset pupils appear from School 15 but not School 7.

```
#Computing SAS values for the Late Testers dataset on correct age (calculated)

JSTLT$RawScore <- factor(JSTLT$RawScore,
                         levels = 0:50,
                         ordered = TRUE)

predicted_raw_prob_LT <- predict(olm, newdata = JSTLT, type = "probs")

predicted_cumprob_LT <- t(apply(predicted_raw_prob_LT, 1, cumsum))

raw_numeric_LT <- as.numeric(as.character(JSTLT$RawScore))

cumprob_obs_score_LT <- predicted_cumprob_LT[cbind(1:nrow(JSTLT), raw_numeric_LT + 1)]

SAS_LT_Computed <- qnorm(cumprob_obs_score_LT, mean = 100, sd = 15)

SAS_LT_Computed[SAS_LT_Computed < 69]  <- 69
SAS_LT_Computed[SAS_LT_Computed > 141] <- 141
SAS_LT_Computed <- round(SAS_LT_Computed)

JSTLT$SAS <- SAS_LT_Computed
```

```
mean(JSTLT$SAS)
```

```
## [1] 100.7789
```

```
sd(JSTLT$SAS)
```

```
## [1] 13.62115
```

```r
#Merging both Datasets
names(JSTMT)
```

```
##  [1] "SchoolID" "PupilID"  "Gender"   "DoB"      "DoT"      "EAL"
##  [7] "FSM"      "RawScore" "age"      "CumProb"  "SAS"
```

```r
names(JSTLT)
```

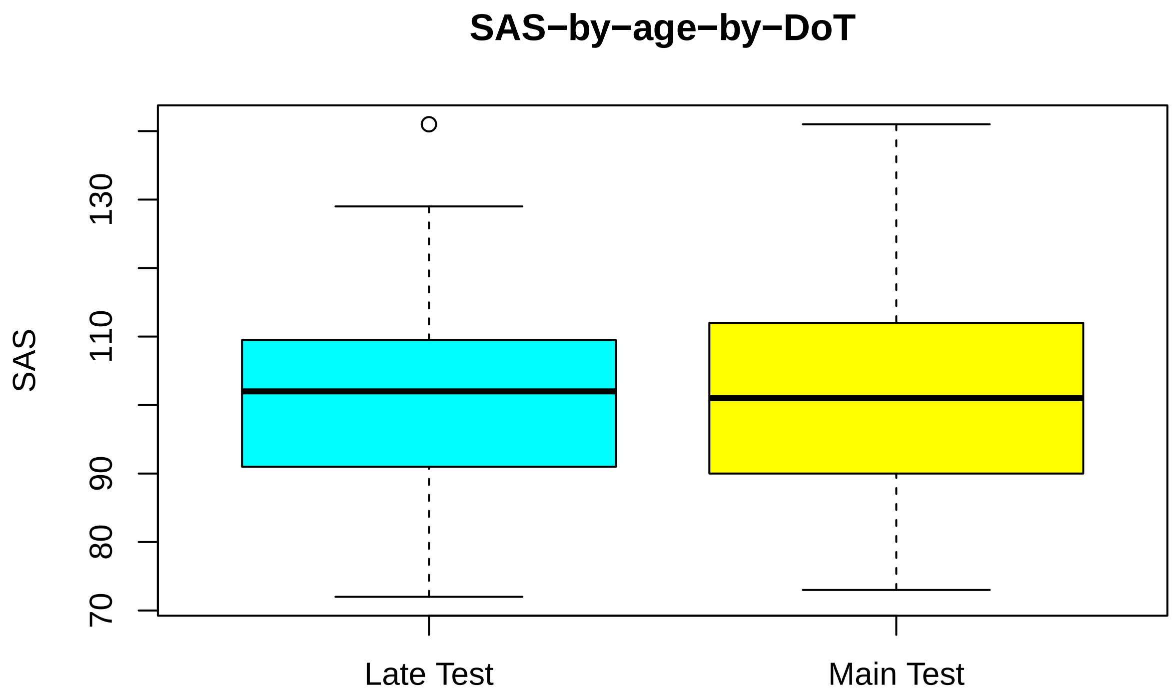
```
##  [1] "SchoolID" "PupilID"  "Gender"   "DoB"      "DoT"      "EAL"
##  [7] "FSM"      "RawScore" "age"      "SAS"
```

```r
JSTMT$CumProb <- NULL
JSTLT <- JSTLT[, names(JSTMT)]

JSTMT$TestGroup <- "Main Test"
JSTLT$TestGroup <- "Late Test"

Final_Dataset <- rbind(JSTMT, JSTLT)
rownames(Final_Dataset) <- NULL
```

```r
boxplot(SAS ~ TestGroup,
        data = Final_Dataset,
        ylab = "SAS",
        xlab = "",
        col = c("cyan", "yellow"),
        main = "SAS-by-age-by-DoT")
```



(15)

- The small differences between the SAS boxplots are explained by natural sampling variation between the Main and Late tester groups.

```
Final_Dataset <- subset(Final_Dataset, select = -TestGroup )
Final_Dataset$SchoolID <- as.numeric(Final_Dataset$SchoolID)
Final_Dataset <- Final_Dataset[order(Final_Dataset$SchoolID,
                                      -Final_Dataset$SAS), ]
rownames(Final_Dataset) <- NULL
```

(16)

- Base R was required so that the work is fully reproducible on any system without relying on external packages.

(17)

- Other factors were not included because the sample sizes within many subgroups (FSM, EAL, SchoolID levels) may be too small to support a multiple predictor ordinal logistic model. This would lead to unstable parameters and produce unreliable cumulative probabilities.
- SAS is designed to adjust only for developmental differences linked to age. Including demographic variables could remove meaningful performance variation and change the interpretation of the score.

- Even if the sample size were large enough to support these additional predictors, SAS must represent age standardised ability, not ability adjusted for demographic characteristics in this instance.

(18)

- Convert each pupil's raw score to an empirical percentile and map it directly onto the N(100,15) scale, avoiding any modelling assumptions.
- Item Response Theory (IRT): Model the probability of each item being answered correctly as a function of pupil ability, estimated ability scores can then be rescaled to SAS.
- Generalised Additive Models (GAM): Allow flexible non-linear relationships between age and raw score, producing smoother age adjusted ability estimates.
- Linear regression standardisation: Use linear modelling of raw score on age (and potentially other predictors) and transform residuals to the N(100,15) scale.
- Machine Learning models (boosted trees)/ other classification ML algorithms

```
write.csv(Final_Dataset, "Zohair_Final_Dataset.csv", row.names = FALSE, quote = FALSE)
```