Stats 369 A4

Richard Choi

09/10/2021

Question 1

Build a classifier to predict labels r from x with xgboost, and show the confusion matrix (You will need to specify the objective function for multi-class prediction, and you will need to remove observations with missing label)

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.1.3
```

```
load("yrbs.rda")
summary(x)
```

```
##        q6              q7               q8                 q9
##  Min.   :1.270   Min.   : 33.57   Length:15624       Length:15624
##  1st Qu.:1.600   1st Qu.: 55.79   Class :character   Class :character
##  Median :1.680   Median : 63.50   Mode  :character   Mode  :character
##  Mean   :1.687   Mean   : 67.78
##  3rd Qu.:1.750   3rd Qu.: 75.75
##  Max.   :2.110   Max.   :180.99
##  NA's   :1266    NA's   :1266
##      q10                q11                q12                q13
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q14                q15                q16                q17
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q18                q19                q20                q21
##  Length:15624       Length:15624       Length:15624       Length:15624
```

```
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q22                q23                q24                q25
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q26                q27                q28                q29
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q30                q31                q32                q33
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q34                q35                q36                q37
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q38                q39                q40                q41
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      q42                q43                q44                q45
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
```

```
##       q46                q47                q48                q49
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q50                q51                q52                q53
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q54                q55                q56                q57
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q58                q59                q60                q61
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q62                q63                q64                q65
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q66                q67                q68                q69
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q70                q71                q72                q73
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
```

```
##
##
##       q74                q75                q76                q77
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q78                q79                q80                q81
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q82                q83                q84                q85
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q86                q87                q88                q89
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q90                q91                q92                q93
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q94                q95                q96                q97
##  Length:15624       Length:15624       Length:15624       Length:15624
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##       q98                q99
##  Length:15624       Length:15624
##  Class :character   Class :character
##  Mode  :character   Mode  :character
```

```
##
##
##
##
```

```r
summary(r)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.00    4.00    4.00    4.27    5.00    7.00     358
```

```r
dim(x)
```

```
## [1] 15624    94
```

```r
dim(r)
```

```
## NULL
```

```r
# predictors
x = x[!is.na(r),]

# label
r = r[!is.na(r)]

x = apply(x, 2, as.numeric)
x = as.matrix(x)

set.seed(369)
xgb.cv(data=x, label=r, num_class=8, nrounds=30, nfold=5, objective="multi:softmax", metrics="merror")
```

```
## [1]  train-merror:0.448284+0.003202  test-merror:0.491813+0.006761
## [2]  train-merror:0.426978+0.004862  test-merror:0.480418+0.008250
## [3]  train-merror:0.411175+0.003931  test-merror:0.473604+0.006668
## [4]  train-merror:0.398336+0.002583  test-merror:0.469345+0.004498
## [5]  train-merror:0.386627+0.002858  test-merror:0.465415+0.002969
## [6]  train-merror:0.377718+0.002943  test-merror:0.462597+0.004510
## [7]  train-merror:0.368236+0.002070  test-merror:0.460895+0.005539
## [8]  train-merror:0.358771+0.002602  test-merror:0.461287+0.006311
## [9]  train-merror:0.349748+0.002481  test-merror:0.457749+0.005568
## [10] train-merror:0.340069+0.001637  test-merror:0.456898+0.006657
## [11] train-merror:0.331734+0.003033  test-merror:0.455588+0.005780
## [12] train-merror:0.322088+0.002055  test-merror:0.454214+0.008664
## [13] train-merror:0.313606+0.002802  test-merror:0.452904+0.008164
## [14] train-merror:0.304484+0.003305  test-merror:0.452249+0.007396
## [15] train-merror:0.295706+0.003231  test-merror:0.451921+0.008292
## [16] train-merror:0.287518+0.003751  test-merror:0.451921+0.007827
## [17] train-merror:0.280787+0.003084  test-merror:0.451658+0.009076
## [18] train-merror:0.273844+0.003557  test-merror:0.451854+0.008960
## [19] train-merror:0.265590+0.003897  test-merror:0.451854+0.009101
## [20] train-merror:0.257975+0.003587  test-merror:0.451264+0.007599
## [21] train-merror:0.251244+0.003981  test-merror:0.451395+0.007706
```

```
## [22] train-merror:0.244432+0.004776  test-merror:0.451002+0.007241
## [23] train-merror:0.238422+0.005867  test-merror:0.451068+0.007602
## [24] train-merror:0.232936+0.004114  test-merror:0.452377+0.008873
## [25] train-merror:0.226058+0.003970  test-merror:0.449952+0.008956
## [26] train-merror:0.220473+0.004783  test-merror:0.450805+0.006190
## [27] train-merror:0.214480+0.003960  test-merror:0.449167+0.006661
## [28] train-merror:0.208797+0.002219  test-merror:0.449102+0.007562
## [29] train-merror:0.203606+0.001828  test-merror:0.448119+0.008034
## [30] train-merror:0.198251+0.001630  test-merror:0.448382+0.008767
```

```
model = xgboost(data=x, label=r, num_class=8, nrounds=12, nfold=5, objective="multi:softmax")
```

```
## [22:27:01] WARNING: amalgamation/../src/learner.cc:627:
## Parameters: { "nfold" } might not be used.
##
##   This could be a false alarm, with some parameters getting used by language bindings but
##   then being mistakenly passed down to XGBoost core, or some parameter actually being used
##   but getting flagged wrongly here. Please open an issue if you find any such cases.
##
##
## [1]  train-mlogloss:1.732108
## [2]  train-mlogloss:1.566009
## [3]  train-mlogloss:1.451141
## [4]  train-mlogloss:1.364378
## [5]  train-mlogloss:1.296632
## [6]  train-mlogloss:1.243450
## [7]  train-mlogloss:1.199090
## [8]  train-mlogloss:1.159675
## [9]  train-mlogloss:1.125581
## [10] train-mlogloss:1.095017
## [11] train-mlogloss:1.067039
## [12] train-mlogloss:1.040387
```

```
pred = predict(model, x)

confMat = table(prediction=pred, true=r)
confMat
```

```
##           true
## prediction    0    1    2    3    4    5    6    7
##          0   38    0    0    0    0    0    0    0
##          1    3  213    7    2   10   10   13   10
##          2    6   25 1010   10  117  133  159   83
##          3    0    0    0   25    0    0    0    0
##          4   83  226  344   35 6332  699 1016  421
##          5   18  107  158   12  204 1158  395   60
##          6   15   56  147   16  186  365 1173   97
##          7    0    0    1    0    0    0    0   68
```

```
sum(diag(confMat))/sum(confMat)
```

```
## [1] 0.656164
```

There doesn't seem to be a significant drop after round 12 so let's use round 12. The accuracy produced by cross validated xgboost seems to be 65.61% which isn't all that great.

Question 2

Describe and visualise which variables are most important in the prediction.

```
importance = xgb.importance(model=model)
importance[which.max(importance$Gain)]
```

```
##    Feature      Gain      Cover  Frequency
## 1:     q97 0.1522095 0.07588264 0.02292388
```

```
importance[which.max(importance$Cover)]
```

```
##    Feature      Gain      Cover  Frequency
## 1:     q97 0.1522095 0.07588264 0.02292388
```

```
importance[which.max(importance$Frequency)]
```

```
##    Feature       Gain      Cover  Frequency
## 1:      q7 0.02949457 0.03983103 0.06271626
```

```
head(importance)
```

```
##    Feature       Gain      Cover  Frequency
## 1:     q97 0.15220945 0.07588264 0.02292388
## 2:      q9 0.09829152 0.03160151 0.01254325
## 3:      q6 0.04668254 0.05008456 0.04930796
## 4:      q8 0.04569027 0.04048740 0.01794983
## 5:     q99 0.04451228 0.03385455 0.01275952
## 6:      q7 0.02949457 0.03983103 0.06271626
```

Gain: The average loss reduction gained when using a predictor in a split.

Cover: The number of times a predictor is used as split; weighted by the number of observations that go through the split.

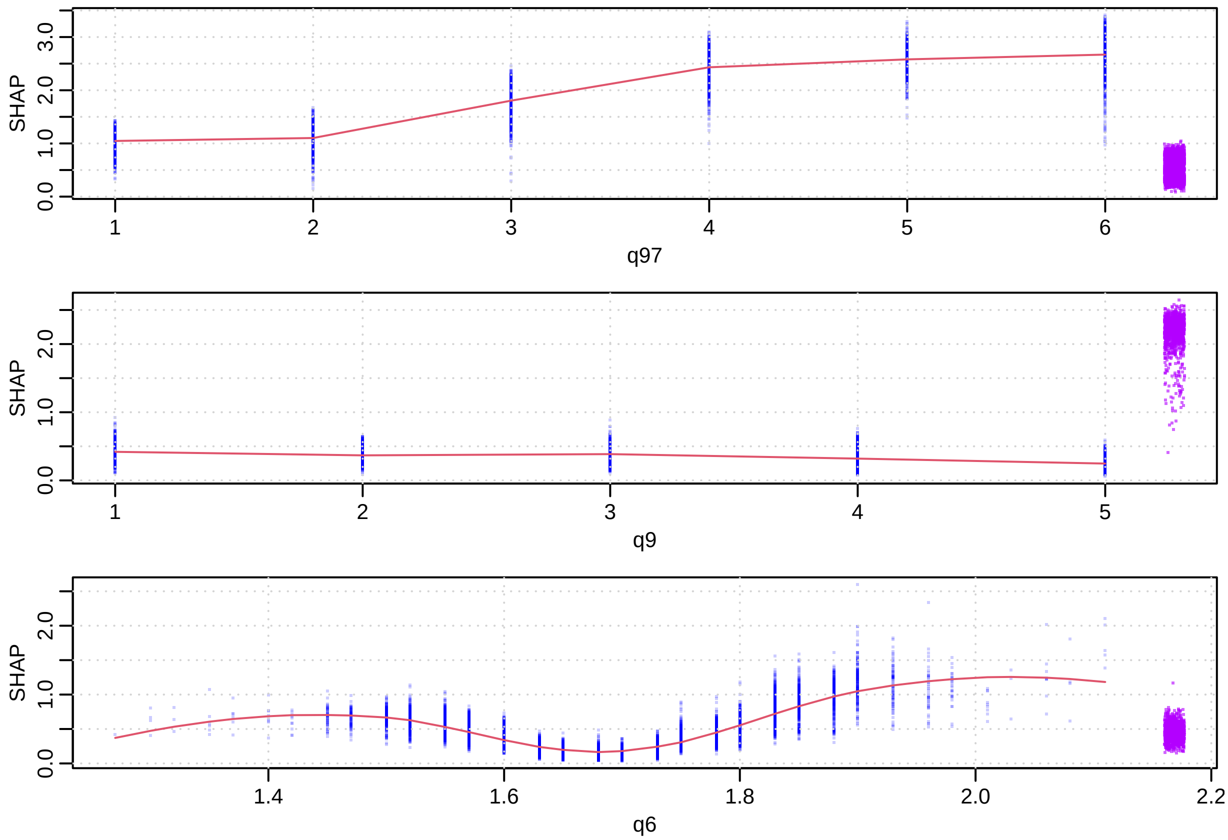
Frequency: The number of times a predictor is used as a (tree) split (across all trees in ensemble method).

Using Gain, Cover, and Frequency it seems that the results are not consistent. Frequency metric shows the predictor q7 to be the most important as it has the highest number of times a predictor is used as a split with 6.27%. However, both gain (15.22%) and cover (7.59%) shows that q97 to be the most important predictor as it has the highest value. This means that q97 reduces the multi class error in a split the most and most used predictor weighted by the number of observations that go through the split.

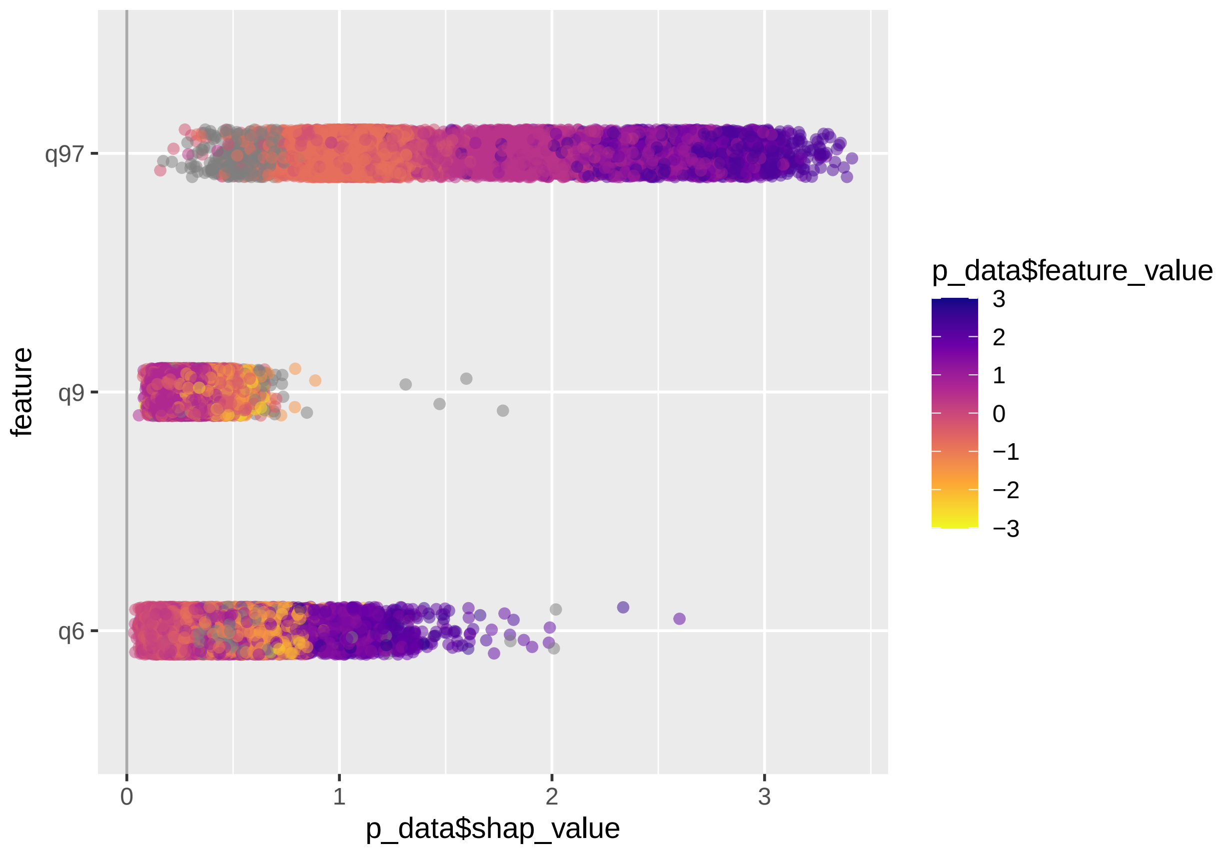
Therefore, we should try using SHAP to make consistent result.

```
xgb.plot.shap(model=model, data=as.matrix(x),top_n=3)
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.975

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 2.025

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4.1006

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.98

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1.02

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1.0404
```




```
xgb.plot.shap.summary(model=model, data=as.matrix(x), top_n=3)
```



Q97: During the past 12 months, how many times have you had a sunburn? It looks like for this question SHAP value seems to increase as there are more times the high school student had a sunburn. Q9: How often do you wear a seat belt when riding in a car driven by someone else? The SHAP value seems to decrease as the frequency of high school student wearing seat belt when riding in a car driven by someone else. Q6: How tall are you without your shoes on? The SHAP value seems to show bi modal relationship in an increasing trend. The SHAP value seems to drop around 1.7 unit peak around 2.0 unit

Using the SHAP values, we have identified Q97, Q9, and Q6 to be the most important predictors as well. By SHAP value, Q97 is the most important then Q6, and Q9 is ranked last. # Question 3 Describe and display the relationships between the most important variables and the label categories – which category/categories is each of the most important variables useful for predicting? Can you produce a summary of the most distinctive predictors for each label category?

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:xgboost':
##
##     slice

## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

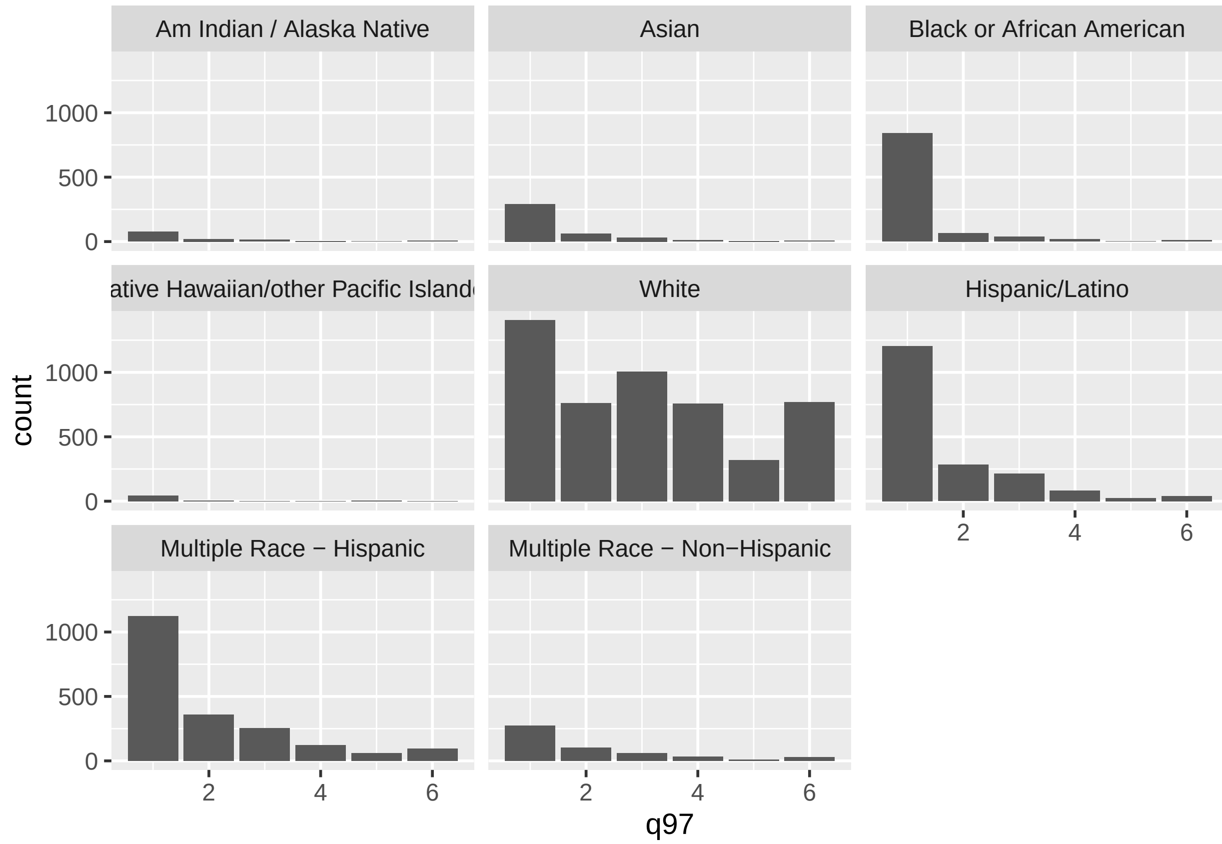
```
full.df = cbind(as.data.frame(x), r)
full.df  = full.df %>%
  mutate(race = factor(r, labels=c("Am Indian / Alaska Native", "Asian", "Black or African American", "I
  select(-r)

full.df %>%
  group_by(race) %>%
  summarise(n())
```

```
## # A tibble: 8 x 2
##   race                                    `n()`
##   <fct>                                   <int>
## 1 Am Indian / Alaska Native                 163
## 2 Asian                                     627
## 3 Black or African American                1667
## 4 Native Hawaiian/other Pacific Islander    100
## 5 White                                    6849
## 6 Hispanic/Latino                          2365
## 7 Multiple Race - Hispanic                 2756
## 8 Multiple Race - Non-Hispanic              739
```

Before we find relationships between variables and label categories let's have a look at the proportion of race in the data. We notice that White, Multiple Race - Hispanic, and Hispanic/Latino are the majority of the race so we will keep that in mind.

```
full.df %>%
  ggplot(aes(x=q97)) + facet_wrap(~race) + geom_bar()
```

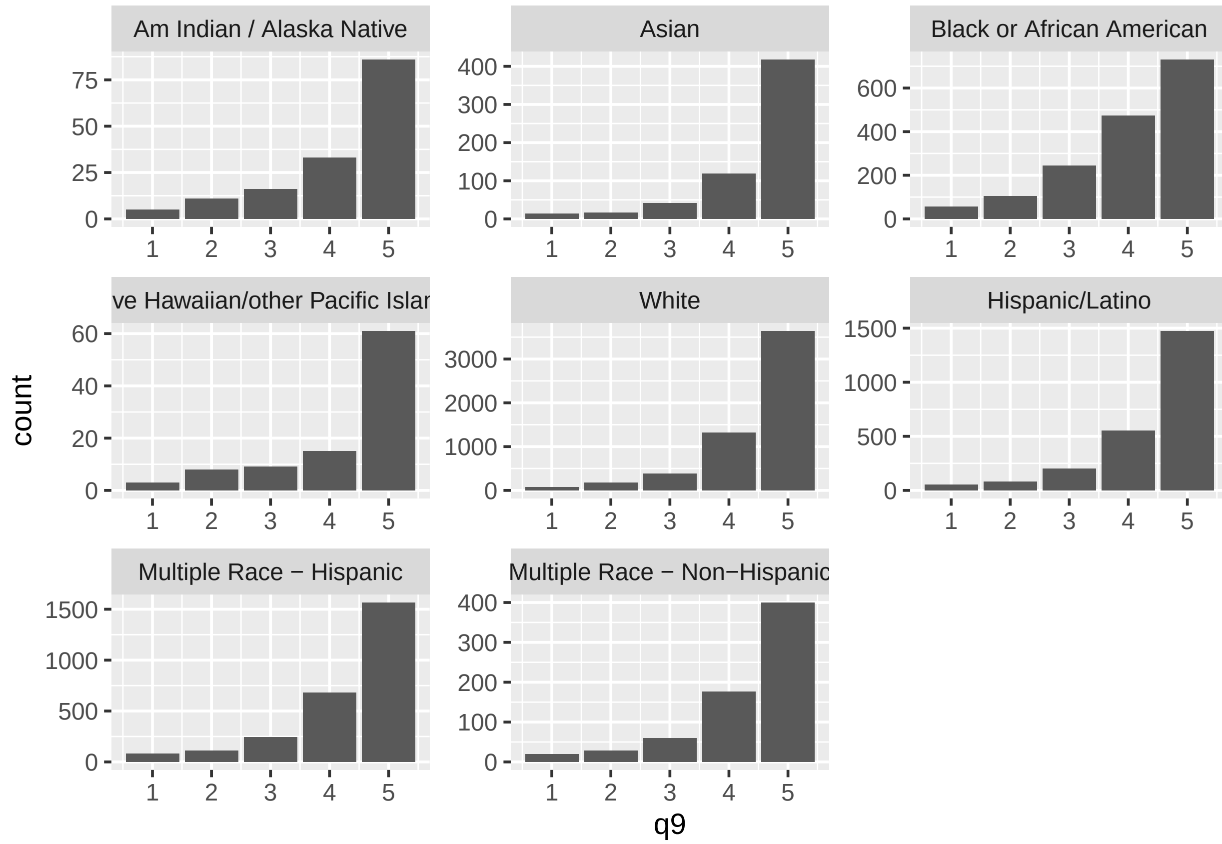
```
## Warning: Removed 4299 rows containing non-finite values (stat_count).
```



We observe that the White people are most prune to sun-burnt followed by people with multiple Hispanic race. Whereas other race like American Indian, Asian, Black, Pacific Islanders are much less prune to sunburn. Based on the plot, there are no Hawaiian/Pacific Islanders who got sunburn. Hispanic and Multiple Race - Hispanic people showed similar distribution so the predictor was not applicable. Therefore, Q97 excels at determining White people and Hawaiian/Pacific Islanders.

```
full.df %>%
  ggplot(aes(x=q9)) + facet_wrap(~race, scales="free") + geom_bar()
```

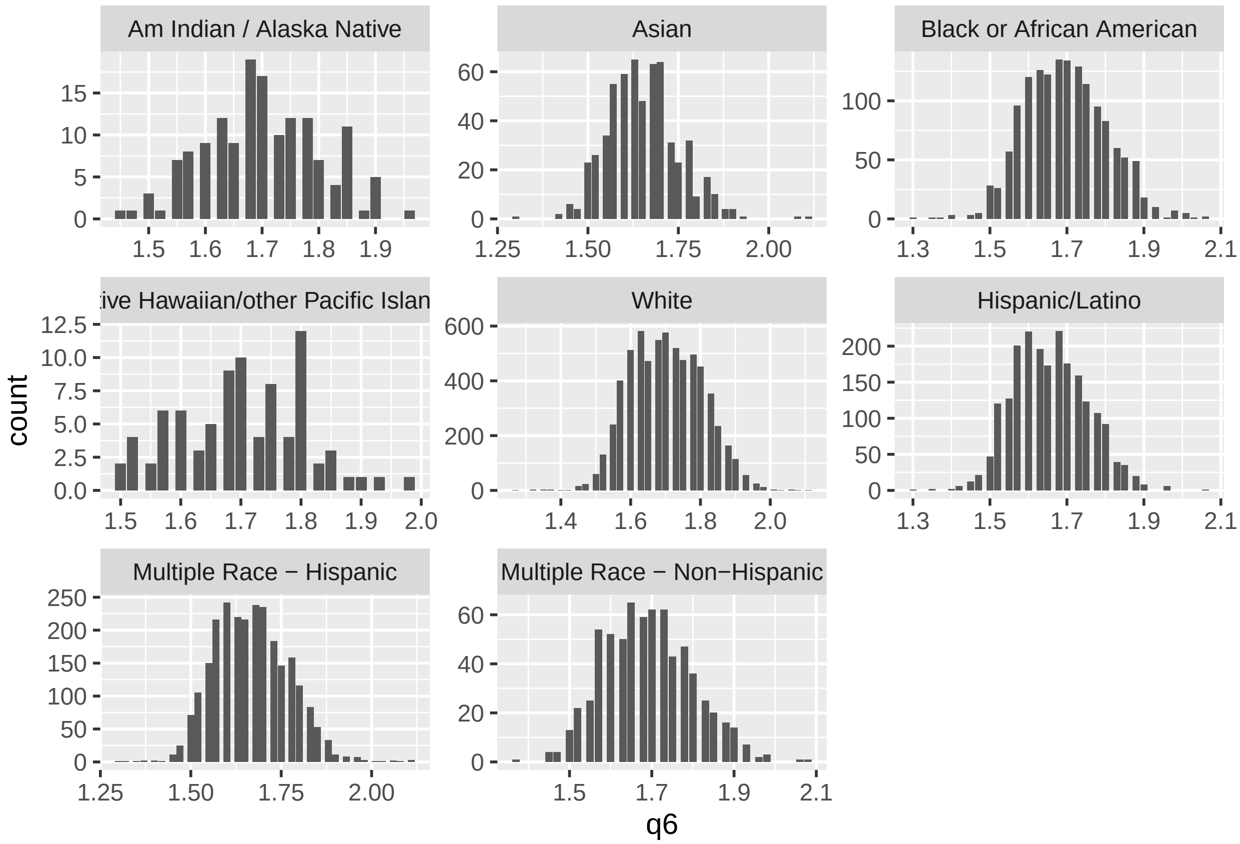
```
## Warning: Removed 1497 rows containing non-finite values (stat_count).
```



By proportion it looks like majority of people regardless of their race always wear their seat belt. We can see that people with black race tend to wear 'most of the time' particularly out of all the races. Q9 is a useful predictor to determine people with black race.

```
full.df %>%
  ggplot(aes(x=q6)) + facet_wrap(~race, scales="free") + geom_bar()
```

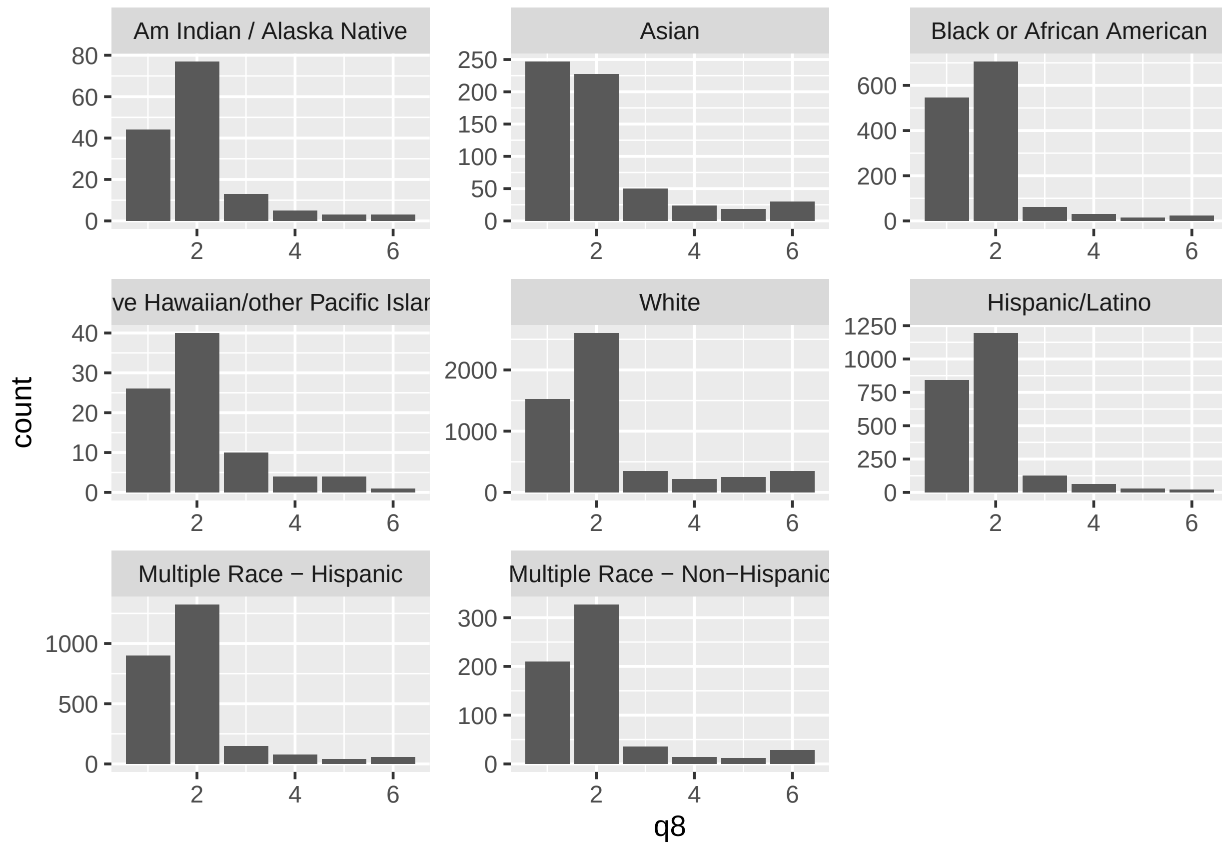
```
## Warning: Removed 1140 rows containing non-finite values (stat_count).
```



Again by proportion, it looks like regardless of race the height of high school students without shoes on are normally distributed. We notice only people with Hawaiian/Other Pacific Islander's height without shoes on drop sharply after 1.8m. Q6 excels at determining people with Hawaiian/Pacific Islander background.

```
full.df %>%
  ggplot(aes(x=q8)) + facet_wrap(~race, scales="free") + geom_bar()
```

```
## Warning: Removed 2335 rows containing non-finite values (stat_count).
```

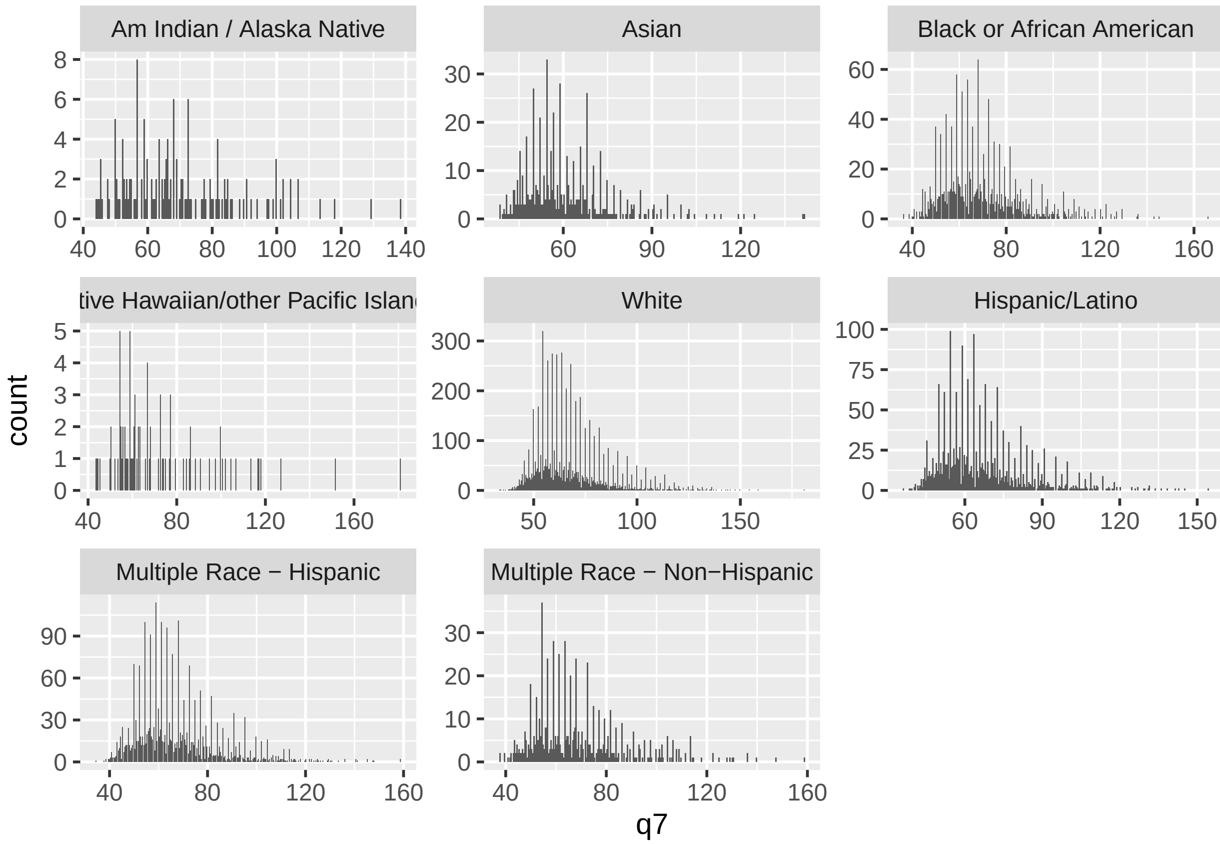


Question 8: When you rode a bicycle during the past 12 months, how often did you wear a helmet?

Again by proportion, it looks like majority of people do not wear a helmet when riding a bicycle. We notice only Asian people in majority do not ride bicycle. Using the data, Q8 excels at determining people with Asian race.

```
full.df %>%
  ggplot(aes(x=q7)) + facet_wrap(~race, scales="free") + geom_bar()
```

```
## Warning: Removed 1140 rows containing non-finite values (stat_count).
```




```
full.df %>%
  group_by(race) %>%
  summarise(average = mean(q7, na.rm=TRUE))
```

```
## # A tibble: 8 x 2
##   race                                 average
##   <fct>                                  <dbl>
## 1 Am Indian / Alaska Native               69.9
## 2 Asian                                   61.0
## 3 Black or African American               69.5
## 4 Native Hawaiian/other Pacific Islander  73.0
## 5 White                                   68.0
## 6 Hispanic/Latino                         67.1
## 7 Multiple Race - Hispanic                67.7
## 8 Multiple Race - Non-Hispanic            68.7
```

Question 7 is "How much do you weigh without your shoes on? (Note: Data are in kilograms." It looks like Asians have the least weight and Pacific Islander have the highest weight on average. Question 7 is an okay predictor to determine Asians and Pacific Islander. However, it needs to take into an account where mean was used so extreme values and different number of race may influence mean values.

In summary, Q97 is good at determining White and Hawaiian/Pacific Islander, Q7 is good at determining Asian and Hawaiian/Pacific Islander, Q8 is good at determining Asian, Q6 is good at determining Hawaiian/Pacific Islander, and Q9 is good at determining black people. Hispanic and multiple race - Hispanic always showed similar distribution so predictors could not be used to determine them. This is perhaps because people with multiple background including Hispanic may share similar culture with people with sole Hispanic background. However, this is only assumption and further researches should be conducted.

There wasn't any distinctive predictor for people with Multiple race without Hispanic background. This could be due to their race's nature. People with White + Asian background and people with Black + Indian American will all be labelled as multiple race without Hispanic background so all their unique characteristic will be jumbled up. Therefore, it is difficult for the predictors to distinguish this label between other races.

Furthermore, people with Am Indian/Alaska Native were one of the minority of race in this data set so it was hard to find any distinctive distribution using the predictors.

Finally as Gain, Cover, and SHAP value recommended, q97 is the best predictor as it distinctively determine people with white race and Hawaiian.

Question 4

Comment on whether (or not) task 3 would be ethically problematic if intended to be published, and for what reasons.

It would be a ethically problematic as the predictors we picked are based on a model with a very poor accuracy. The accuracy is only 64.19% which is very low. Moreover, the task involves using race's characteristic or activities which may reinforce stereotypes. For example, we found that people with white background will have a high chance of sunburn and people with Pacific background tend to be overweight. Publicising could therefore be a problem in a wider society. For instance, using this information the insurance companies may impose higher premium to people with black ground for not wearing seat belts or may cause racial hate using these difference in racial characteristics.