Assignment 3[*]

PSTAT 231

Villaseñor-Derbez J.C. | 8749749

1 Set up

```
# Load packages
suppressPackageStartupMessages({
  library(startR)
  library(here)
  library(magrittr)
  library(tree)
  library(maptree)
  library(ROCR)
  library(ggridges)
  library(tidyverse)
})
```

```
# Some housekeeping
update_geom_defaults("point", list(fill = "steelblue",
                                   color = "black",
                                   shape = 21,
                                   size = 2))

update_geom_defaults("line", list(color = "black",
                                  size = 1))

update_geom_defaults("density_ridges", list(fill = "steelblue",
                                     color = "black",
                                     size = 1,
                                     alpha = 0.5))

# Set global theme
theme_set(startR::ggtheme_plot())
```

```
# Load the data
drug_use <- read_csv(here("data", "drug.csv"),
                     col_names = c("ID","Age","Gender","Education","Country","Ethnicity",
                                   "Nscore","Escore","Oscore","Ascore","Cscore","Impulsive",
                                   "SS","Alcohol","Amphet","Amyl","Benzos","Caff","Cannabis",
                                   "Choc","Coke","Crack","Ecstasy","Heroin","Ketamine",
                                   "Legalh","LSD","Meth","Mushrooms","Nicotine","Semer","VSA"),
                     col_types = cols())
```

[*] Code available on GitHUb at: https://github.com/jcvdav/PSTAT231/tree/master/docs/assig3

2 Logistic regression for drug use

2.1 Feature engineering

```
# Create ordered factors for alcohol trhoug VSA
drug_use <- drug_use %>%
  mutate_at(as.ordered, .vars=vars(Alcohol:VSA))

# Create orederd factor for gender, ethnicity and country
drug_use <- drug_use %>%
  mutate(Gender = factor(Gender,
                         labels = c("Male",
                                    "Female")),
         Ethnicity = factor(Ethnicity,
                            labels = c("Black",
                                       "Asian",
                                       "White",
                                       "Mixed:White/Black",
                                       "Other",
                                       "Mixed:White/Asian",
                                       "Mixed:Black/Asian")),
         Country = factor(Country,
                          labels = c("Australia",
                                     "Canada",
                                     "New Zealand",
                                     "Other",
                                     "Ireland",
                                     "UK",
                                     "USA")))
```

2.2 Define a new factor response variable `recent_cannabis_use` which is "Yes" if a person has used cannabis within a year, and "No" otherwise. This can be done by checking if the Cannabis variable is greater than or equal to CL3. Hint: use mutate with the ifelse command. When creating the new factor set levels argument to levels=c("No", "Yes") (in that order).

```
drug_use <- drug_use %>%
  mutate(recent_cannabis_use = ifelse(Cannabis != "CL3", "No", "Yes"),
         recent_cannabis_use = factor(recent_cannabis_use,
                                      labels = c("No", "Yes")))
```

2.3 We will create a new tibble that includes a subset of the original variables. We will focus on all variables between age and SS as well as the new factor related to recent cannabis use. Create drug_use_subset with the command:

```
drug_use_subset <- drug_use %>%
  select(Age:SS, recent_cannabis_use)
```

Split drug_use_subset into a training data set and a test data set called drug_use_train and drug_use_test. The training data should include 1500 randomly sampled observation and the test data should include the remaining observations in drug_use_subset. Verify that the data sets are of the right size by printing dim(drug_use_train) and dim(drug_use_test).

```
# set seed
set.seed(42)

# Get rows for training set
train_rows <- sample(x = 1:nrow(drug_use_subset),
                     size = 1500,
                     replace = FALSE)

# Create training set
drug_use_train <- drug_use_subset[train_rows, ]
# Create testing set
drug_use_test <- drug_use_subset[-train_rows, ]

# Check dimensions
dim(drug_use_train)
```

```
## [1] 1500   13
```

```
dim(drug_use_test)
```

```
## [1] 385  13
```

2.4 Fit a logistic regression to model recent_cannabis_use as a function of all other predictors in drug_use_train. Fit this regression using the training data only. Display the results by calling the summary function on the logistic regression object.

```
cannabis_model <- glm(recent_cannabis_use ~ .,
                      data = drug_use_train,
                      family = binomial(link = "logit"))

stargazer::stargazer(cannabis_model,
                     single.row = T,
                     header = F,
                     title = "Logistic regression modelling recent cannabis use as a function of all oth
```

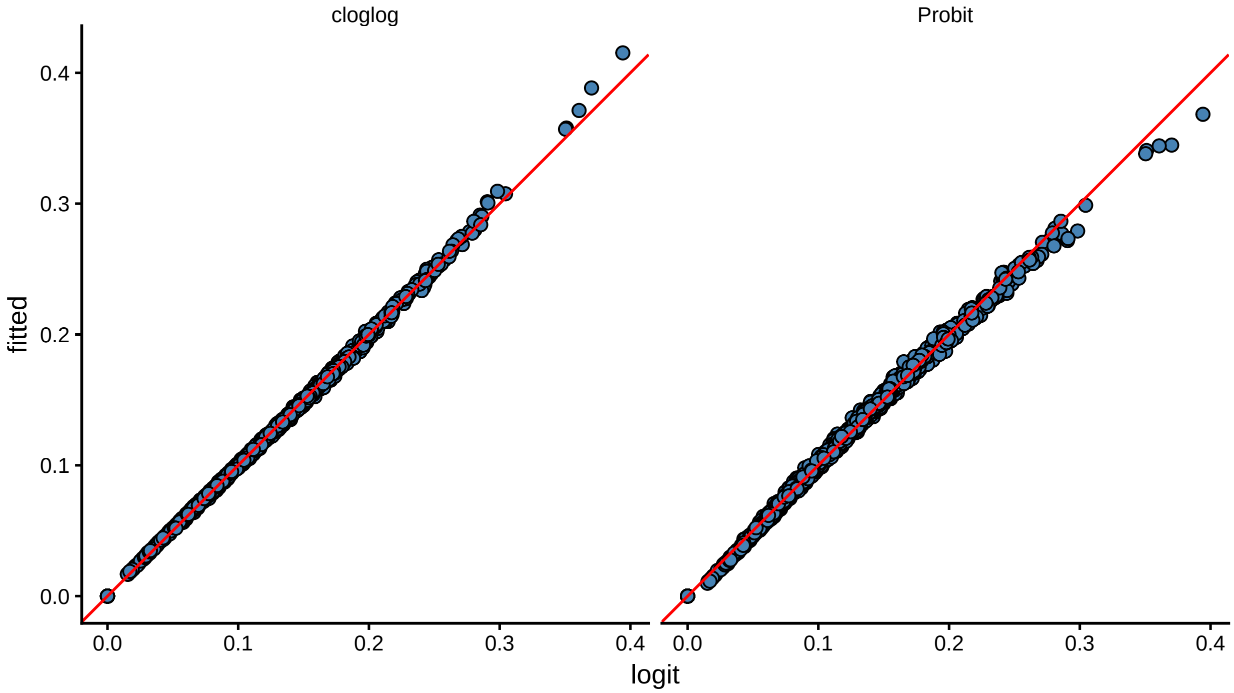
2.5 Probit and c-log-log link functions

```
cannabis_model_probit <- glm(recent_cannabis_use ~ .,
                             data = drug_use_train,
                             family = binomial(link = "probit"))

cannabis_model_cloglog <- glm(recent_cannabis_use ~ .,
                              data = drug_use_train,
                              family = binomial(link = "cloglog"))
```

Table 1: Logistic regression modelling recent cannabis use as a function of all other predictors in the training dataset. Numbers in parentheses are standard errors of the estimates.

	Dependent variable:
	recent_cannabis_use
Age	−0.311*** (0.115)
GenderFemale	−0.061 (0.182)
Education	0.334*** (0.102)
CountryCanada	−14.707 (1,181.812)
CountryNew Zealand	0.184 (0.351)
CountryOther	0.558 (0.431)
CountryIreland	0.276 (0.801)
CountryUK	0.482 (0.401)
CountryUSA	0.247 (0.221)
EthnicityAsian	−13.725 (513.896)
EthnicityWhite	1.086 (1.030)
EthnicityMixed:White/Black	0.280 (1.462)
EthnicityOther	0.844 (1.134)
EthnicityMixed:White/Asian	−13.741 (624.941)
EthnicityMixed:Black/Asian	−13.558 (1,670.080)
Nscore	0.183* (0.101)
Escore	−0.100 (0.101)
Oscore	0.032 (0.098)
Ascore	−0.008 (0.089)
Cscore	−0.188* (0.100)
Impulsive	−0.126 (0.116)
SS	0.236* (0.123)
Constant	−3.328*** (1.039)
Observations	1,500
Log Likelihood	−505.870
Akaike Inf. Crit.	1,057.740
Note:	*p<0.1; **p<0.05; ***p<0.01

```
tibble(logit = cannabis_model$fitted.values,
       Probit = cannabis_model_probit$fitted.values,
       cloglog = cannabis_model_cloglog$fitted.values) %>%
  gather(model, fitted, -logit) %>%
  ggplot(aes(x = logit, y = fitted)) +
  geom_point() +
  facet_wrap(~model) +
  geom_abline(intercept = 0,
              slope = 1,
              color = "red") +
  coord_equal()
```



The c-log-log regression produced fitted values that are more similar to the logistic regression using a logit-link function. For higher probabilities, the c-log-log function produces higher fitted values (above the red line), while the probit function produces lower probabilities (below the read line). The probit link produces smaller probabilities than the logit for intervals 0-0.05 and 0.25 - 0.4, where blue points consistently appear below the red line. The c-log-log link function has also much less variation, while the probit link shows greater variance in the middle.

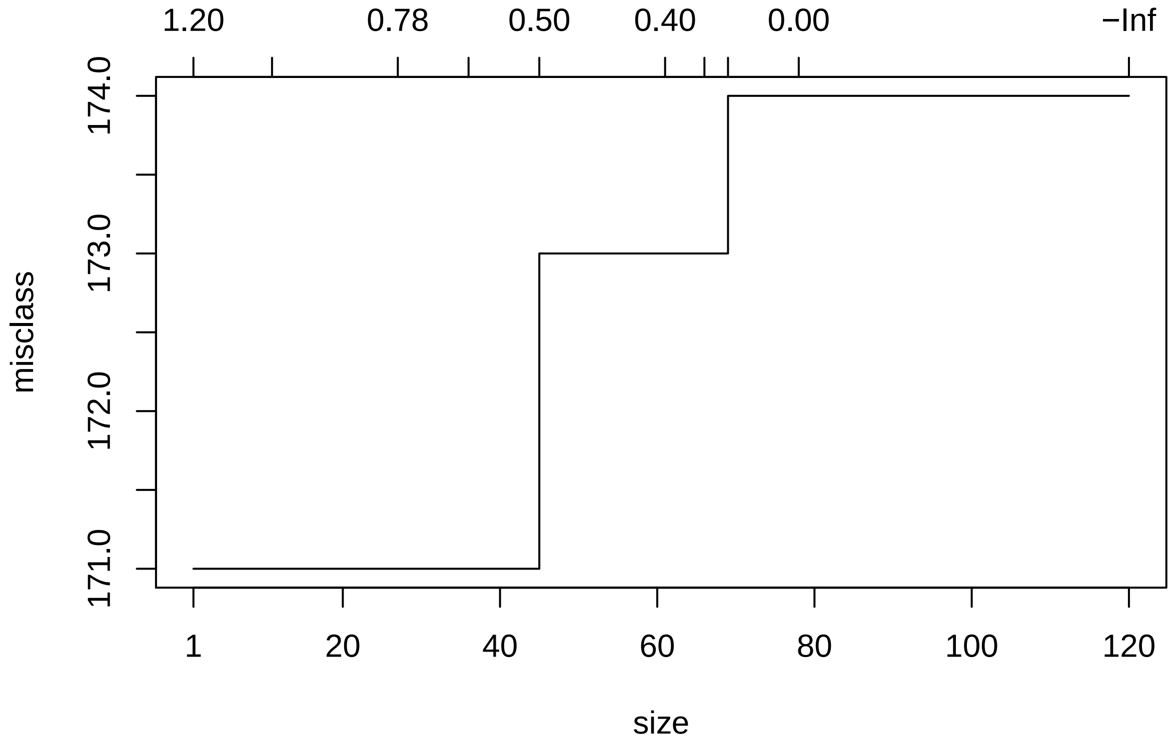
3 Decision Tree for drug use

3.1 Construct a decision tree to predict `recent_cannabis_use` using all other predictors in `drug_use_train`. Set the value of the argument `control = tree_parameters` where tree_parameters are:

```r
tree_parameters <- tree.control(nobs = nrow(drug_use_train),
                                minsize = 10,
                                mindev = 1e-3)
```

3.2 Use 10-fold CV to select the a tree which minimizes the cross-validation misclassification rate. Use the function `cv.tree`, and set the argument `FUN = prune.misclass`. Find the size of the tree which minimizes the cross validation error. If multiple trees have the same minimum cross validated misclassification rate, set `best_size` to the smallest tree size with that minimum rate.

```r
set.seed(43)
# Grow decision tree
drug_tree <- tree(recent_cannabis_use ~ .,
                  data = drug_use_train,
                  control = tree_parameters)

# Cross-validate tree
cv_drug_tree <- cv.tree(object = drug_tree,
                        method = "misclass",
                        K = 10)

plot(cv_drug_tree)
```

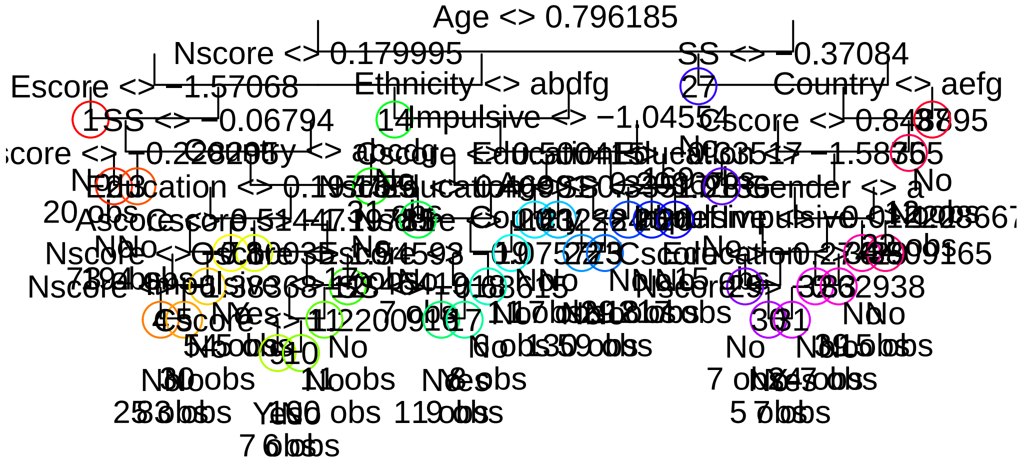


```
# create a tidy version of the diagnostics
cv_drug_tidy <- tibble(size = cv_drug_tree$size,
                       misclass = cv_drug_tree$dev)

# Find the
cv_drug_tidy %>%
  filter(misclass <= min(misclass)) %>%
  filter(size == max(size))
```

```
## # A tibble: 1 x 2
##    size misclass
##   <int>    <dbl>
## 1    36     171.
```

3.3 Prune the tree to the size found in the previous part and plot the tree using the draw.tree function from the maptree package. Set nodeinfo = TRUE. Which variable is split first in this decision tree?

```
drug_tree_pruned <- prune.tree(tree = drug_tree, best = 36)

draw.tree(drug_tree_pruned)
```



3.4 Compute and print the confusion matrix for the test data using the function table(truth, predictions) where truth and predictions are the true classes and the predicted classes from the tree model respectively. Calculate the true positive rate (TPR) and false positive rate (FPR) for the confusion matrix. Show how you arrived at your answer.

```r
truth <- drug_use_test$recent_cannabis_use
predictions <- predict(object = drug_tree_pruned, newdata = drug_use_test, type = "class")

table(truth, predictions)
```

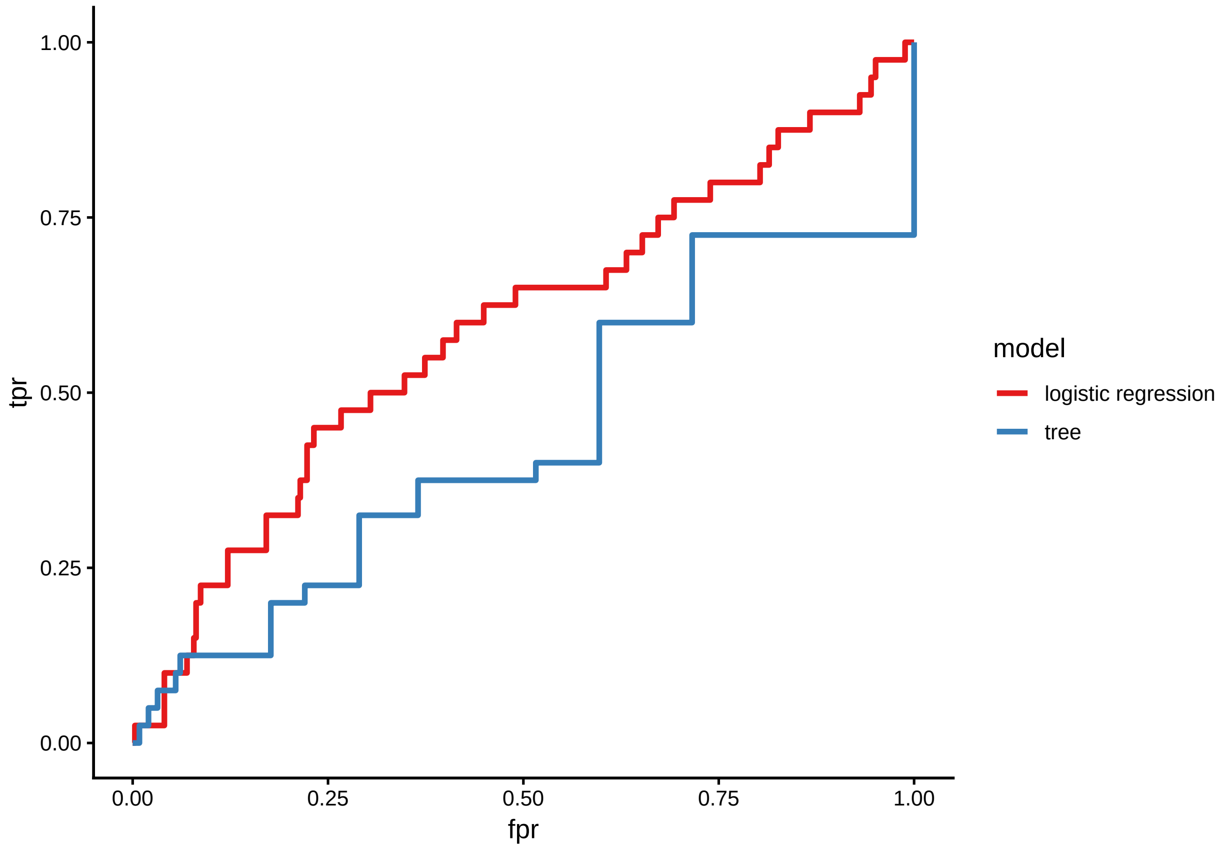
```
##      predictions
## truth  No Yes
##   No  337   8
##   Yes  38   2
```

- TPR is $2/(38 + 2) = 0.05$
- FPR is $8/(337 + 8) = 0.023$

4 Model Comparison

4.1 Plot the ROC curves for both the logistic regression fit and the decision tree on the same plot. Use `drug_use_test` to compute the ROC curves for both the logistic regression model and the best pruned tree model.

```r
# Logistic ROC
log_pred <- prediction(predict(cannabis_model,
                               newdata = drug_use_test,
                               type = "response"), truth)
log_perf <- performance(log_pred, "tpr", "fpr")

# Tree ROC
tree_pred <- prediction(predict(object = drug_tree_pruned,
                                newdata = drug_use_test)[, 2],
                        truth)
tree_perf <- performance(tree_pred, "tpr", "fpr")

# Plot it
tibble(fpr = log_perf@x.values[[1]],
       tpr = log_perf@y.values[[1]],
       model = "logistic regression") %>%
  rbind(tibble(fpr = tree_perf@x.values[[1]],
               tpr = tree_perf@y.values[[1]],
               model = "tree")) %>%
  ggplot(aes(x = fpr, y = tpr, color = model)) +
  geom_step(size = 1) +
  scale_color_brewer(palette = "Set1")
```



4.2 Compute the AUC for both models and print them. Which model has larger AUC?

```
log_auc <- performance(log_pred, "auc")@y.values[[1]]
tree_auc <- performance(tree_pred, "auc")@y.values[[1]]
```

The AUC for logistic is $AUC_{logistic} = 0.59$ and the AUC for the tree is $AUC_{tree} = 0.49$.

5 Clustering and dimension reduction for gene expression data

5.1 The class of the first column of `leukemia_data`, `Type`, is set to character by default. Convert the `Type` column to a factor using the `mutate` function. Use the `table` command to print the number of patients with each leukemia subtype. Which leukemia subtype occurs the least in this data?

```
leukemia_data <- read.csv(here("data", "leukemia_data.csv"),
                          stringsAsFactors = F) %>%
  mutate(Type = factor(Type))
```

```
table(leukemia_data$Type) %>%
  as_tibble() %>%
  arrange(n) %>%
```

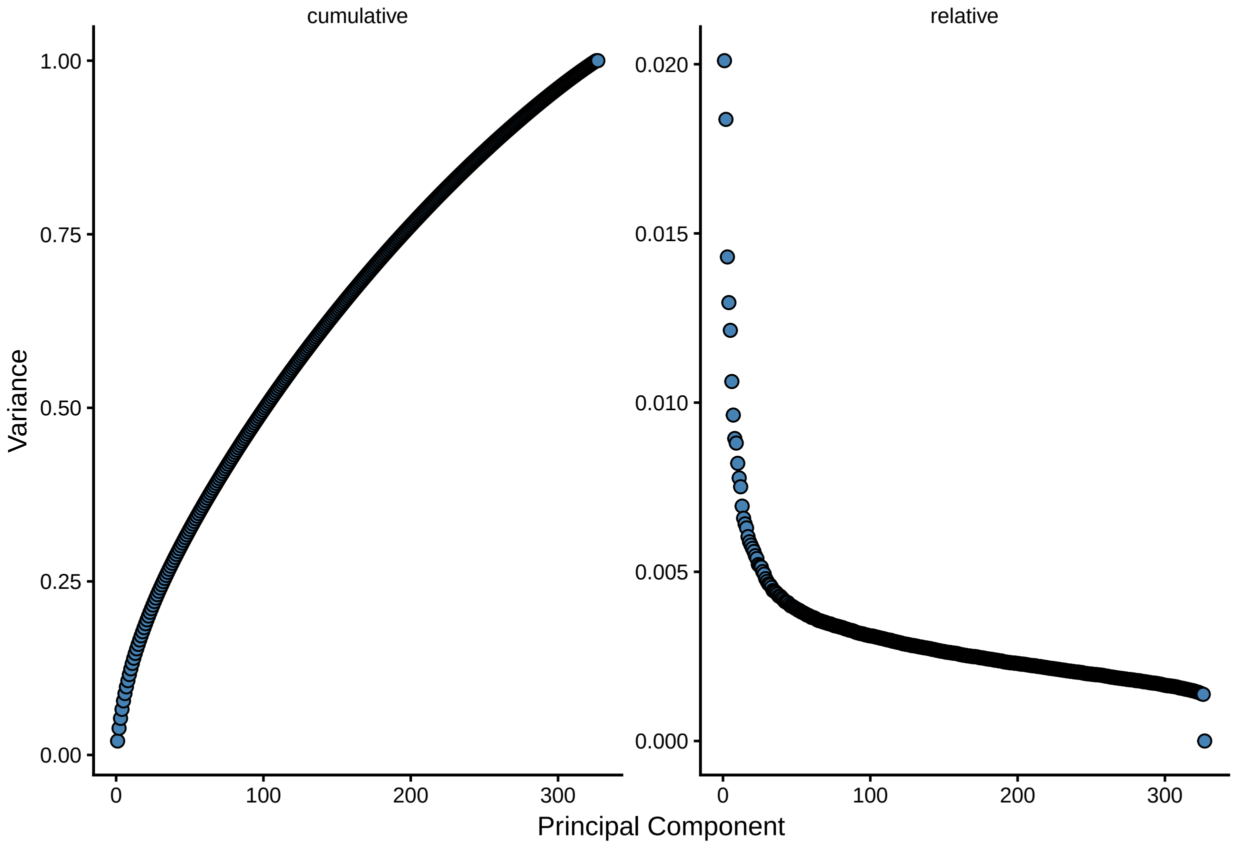
```r
  knitr::kable(booktabs = T,
               col.names = c("Type", "Frequency"))
```

Type	Frequency
BCR-ABL	15
MLL	20
E2A-PBX1	27
T-ALL	43
Hyperdip50	64
OTHERS	79
TEL-AML1	79

The least common leukemia subtype is BCR-ABL, with 15 cases.

5.2 Run PCA on the leukemia data using `prcomp` function with `scale = TRUE` and `center = TRUE` (this scales each gene to have mean 0 and variance 1). Make sure you exclude the `Type` column when you run the PCA function (we are only interested in reducing the dimension of the gene expression values and PCA doesn't work with categorical data anyway). Plot the proportion of variance explained by each principal component (PVE) and the cumulative PVE side-by-side.

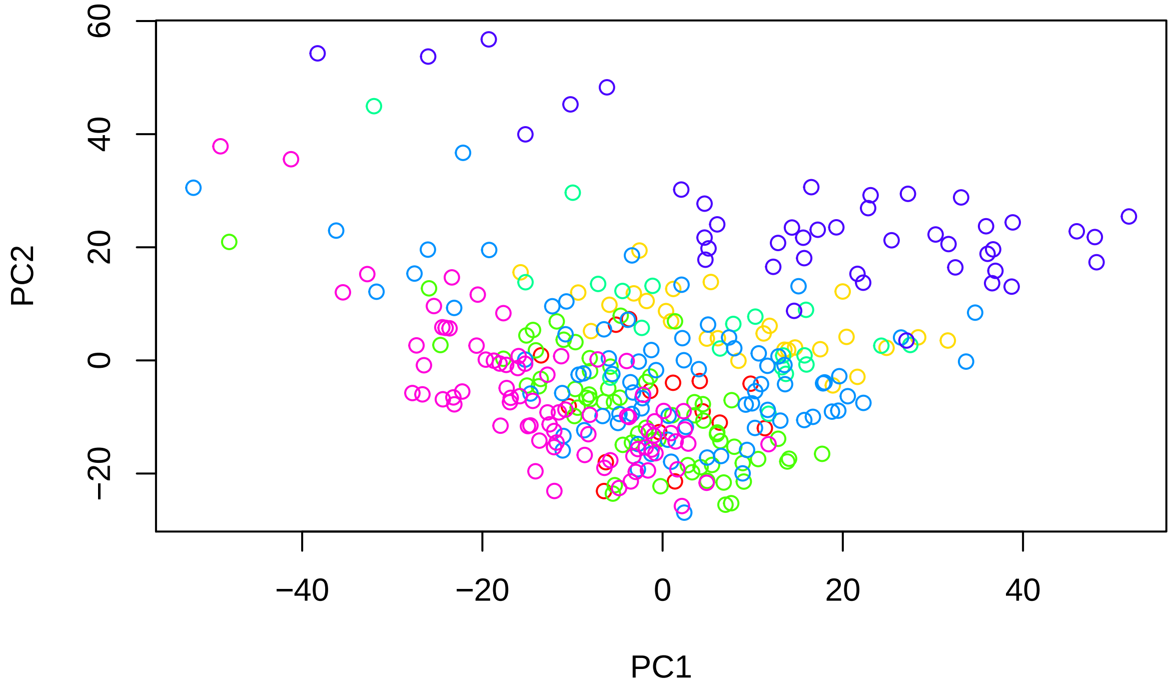
```r
leuk_pca <- leukemia_data %>%
  select(-Type) %>%
  prcomp(scale = T, center = T)

tibble(component = 1:length(leuk_pca$sdev),
       variance = leuk_pca$sdev) %>%
  mutate(relative = variance / sum(variance),
         cumulative = cumsum(relative)) %>%
  select(-variance) %>%
  gather(variance, value, -component) %>%
  ggplot(aes(x = component, y = value)) +
  geom_point() +
  facet_wrap(~variance, scales = "free_y") +
  labs(x = "Principal Component", y = "Variance")
```



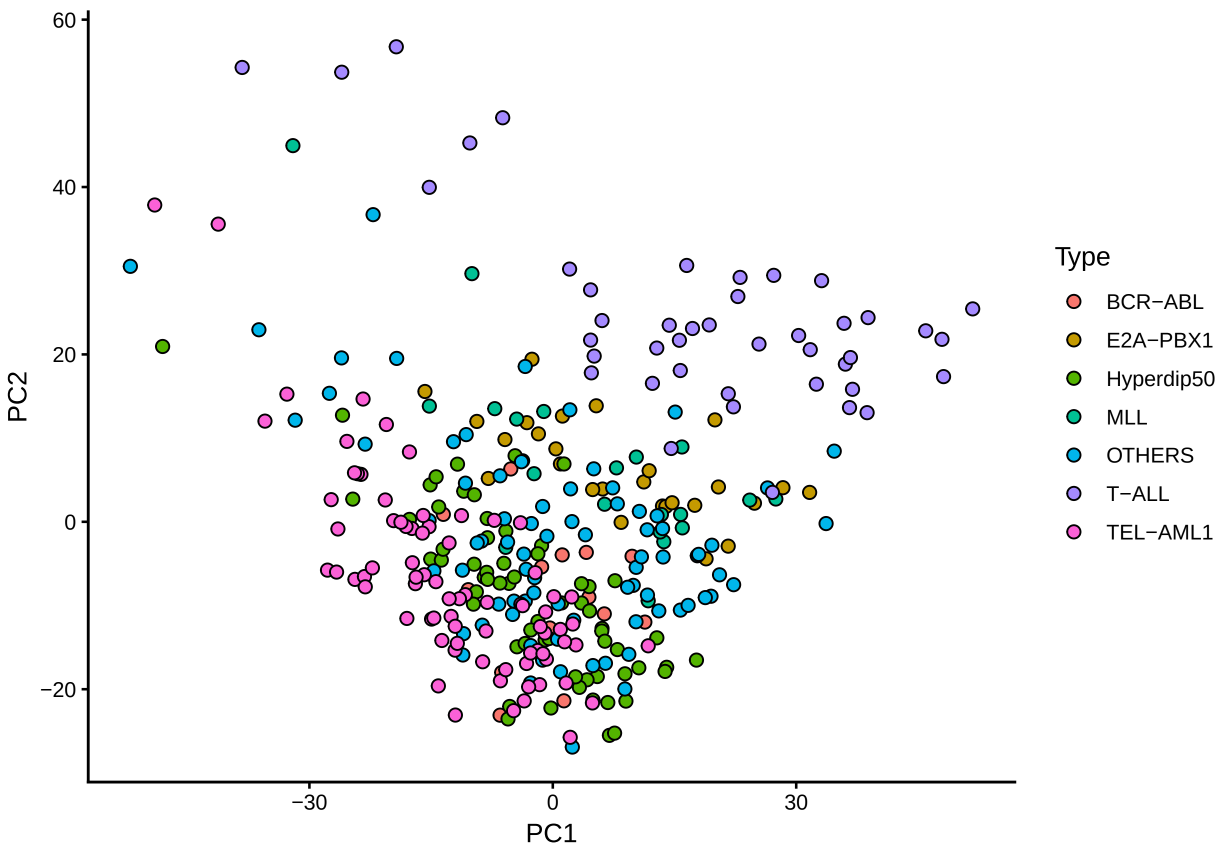
5.3 Use the results of PCA to project the data into the first two principal component dimensions. `prcomp` returns this dimension reduced data in the first columns of `x`. Plot the data as a scatter plot using plot function with `col = plot_colors` where `plot_colors` is defined:

```
rainbow_colors <- rainbow(7)
plot_colors <- rainbow_colors[leukemia_data$Type]
```

```
plot(leuk_pca$x[,1:2], col = plot_colors)
```



```
#Sorry, but I prefer ggplot2
leuk_pca$x %>%
  as_tibble() %>%
  mutate(Type = leukemia_data$Type) %>%
  ggplot(aes(x = PC1, y = PC2, fill = Type)) +
  geom_point()
```


60
40
20
0
−20
PC2
−30
0
30
PC1
Type
BCR−ABL
E2A−PBX1
Hyperdip50
MLL
OTHERS
T−ALL
TEL−AML1