Homework 9

Data Description

We will illustrate the principle of cross validation using the `Default` dataset from the textbook *Introduction to Statistical Learning* by James, Witten, Hastie, and Tibshirani. This dataset contains simulated (but realistic) information about ten thousand bank customers. The aim is to predict which customers will default (Yes or No) on their credit card debt. The variables include:

- `default` : 1 if the customer has defaulted, 0 if not.
- `student` : 1 if the customer is a student, 0 if not.
- `balance` : the average balance that the customer has remaining on their card balance after making their monthly payments.
- `income` : the income of a customer.

First, load the dataset by running the following commands.

```
Default <- read_csv("https://raw.githubusercontent.com/dsnair/ISLR/master/data/csv/Default.csv")
```

```
## Rows: 10000 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (2): default, student
## dbl (2): balance, income
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Default <- Default %>% mutate(default = factor(default, levels = c("No", "Yes")))
```

Problem 1 (3 pts)

Fit a logistic regression model that uses `income` and `balance` to predict `default` . Using this model and a cutoff of a probability of 0.5 for predicting a default, **what is the classification accuracy of the model based on the entire dataset.**

Hint: Recall that to fit a logistic regression, the basic formula is `my_fit <- glm(my_outcome ~ variable1 + variable2 + etc, family = binomial, data = my_data)` . After fitting the model, we can estimate the probability of a success (in this case, not defaulting) by running `predict(my_fit, possibly_different_data, type = 'response')` .

```
# Your R code here
my_fit <- glm(default ~ income + balance, family = binomial, data = Default)
predicted <- predict(my_fit, Default, type = 'response')
predicted_default <- ifelse(predicted > 0.5, "Yes", "No")
mean(predicted_default == Default$default)
```

```
## [1] 0.9737
```

Answer: The classification accuracy of the model based on the entire dataset is 97.37%

Problem 2 (2 pts)

Divide the data into a training and testing set using the following code.

```
library(caret)
set.seed(1123581321)
idx_train <- createDataPartition(Default$default, p = 0.5)[[1]]
Default_train <- Default[idx_train,]
Default_test  <- Default[-idx_train,]
```

Repeat Problem 1, but instead of using the full dataset, use the training set to fit the logistic regression model and the testing set to compute the accuracy. **How does the result compare to what you get when you use the full data? Does the model appear to be overfitting?**

```
# Your R code here.
my_fit <- glm(default ~ income + balance, family = binomial, data = Default_train)
predicted <- predict(my_fit, Default_test, type = 'response')
predicted_default <- ifelse(predicted > 0.5, "Yes", "No")
mean(predicted_default == Default_test$default)
```

```
## [1] 0.9731946
```

Answer: When comparing this probability of 97.32% against the probability when using the full data (97.37%), it can be seen that the results are extremely similar, but slightly less (by 0.05%). The model does not appear to be overfitting as the accuracy on the test data is pretty high and close to the accuracy when using the full data.

Problem 3 (2 pts)

Consider a logistic regression model that predicts the probability of `default` using `income`, `balance`, and `student`. Estimate the accuracy from this model using the same train/test split used in Problem 2. Comment on whether or not including `student` in the model leads to an improvement in the accuracy on the test set.

```
# R code goes here.
my_fit <- glm(default ~ income + balance + student, family = binomial, data = Default_train)
predicted <- predict(my_fit, Default_test, type = 'response')
predicted_default <- ifelse(predicted > 0.5, "Yes", "No")
mean(predicted_default == Default_test$default)
```

```
## [1] 0.9727946
```

Answer: Including student in the model does not lead to an improvement in the accuracy on the test set. It actually decreases by about 0.04%.

Problem 4 (3 pts)

Compare the model with `income`, `balance` and `student` to the model with `income` and `balance` using 20-fold cross validation instead. How does this compare with the train/test split approach? *Hint: you can use `train` in the `caret` package to do this by looking at `my_caret_fit$results`. The recipe for `train` is*
`my_fit <- train(my_outcome ~ variable1 + variable2 + etc, method = "glm", data = my_data, trControl = train_control)`.

Note: this may require you to install the `e1071` package.

Another Note: remember to use the same seed for both caret fits.

```
library(caret)
set.seed(12390)
tr_control <- trainControl(method = "cv", number = 20)

## Your code here.
set.seed(12390)
ibs_fit <- train(default ~ income + balance + student, method = "glm", data = Default, trControl = tr_contro
l)
set.seed(12390)
ib_fit <- train(default ~ income + balance, method = "glm", data = Default, trControl = tr_control)

ibs_fit$results$Accuracy
```

```
## [1] 0.9733021
```

```
ib_fit$results$Accuracy
```

```
## [1] 0.9738021
```

Answer: When using the 20-fold cross validation approach, it can be seen that the classification accuracy of the model using `income` and `balance` is 97.38% which is greater than the classification accuracy of the model using `income`, `balance`, and `student`, which is 97.33%, by 0.05%. This trend/relationship is almost exactly the same as the one exhibited by the train/test split approach, though the difference here is greater by 0.01%. However, the accuracy of each model with the same parameters is slightly different for both approaches as the 20-fold cross validation approach produces a greater accuracy compared to the train/test split approach by about 0.05% or 0.06%. 97.38% > 97.32% and 97.33% > 97.28%.