Practical 5

Emilia Löscher

2022-10-10

First, load the packages:

```
library(MASS)
library(class)
library(ISLR)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
library(magrittr)
```

```
##
## Attache Paket: 'magrittr'
##
## Das folgende Objekt ist maskiert 'package:purrr':
##
##     set_names
##
## Das folgende Objekt ist maskiert 'package:tidyr':
##
##     extract
```

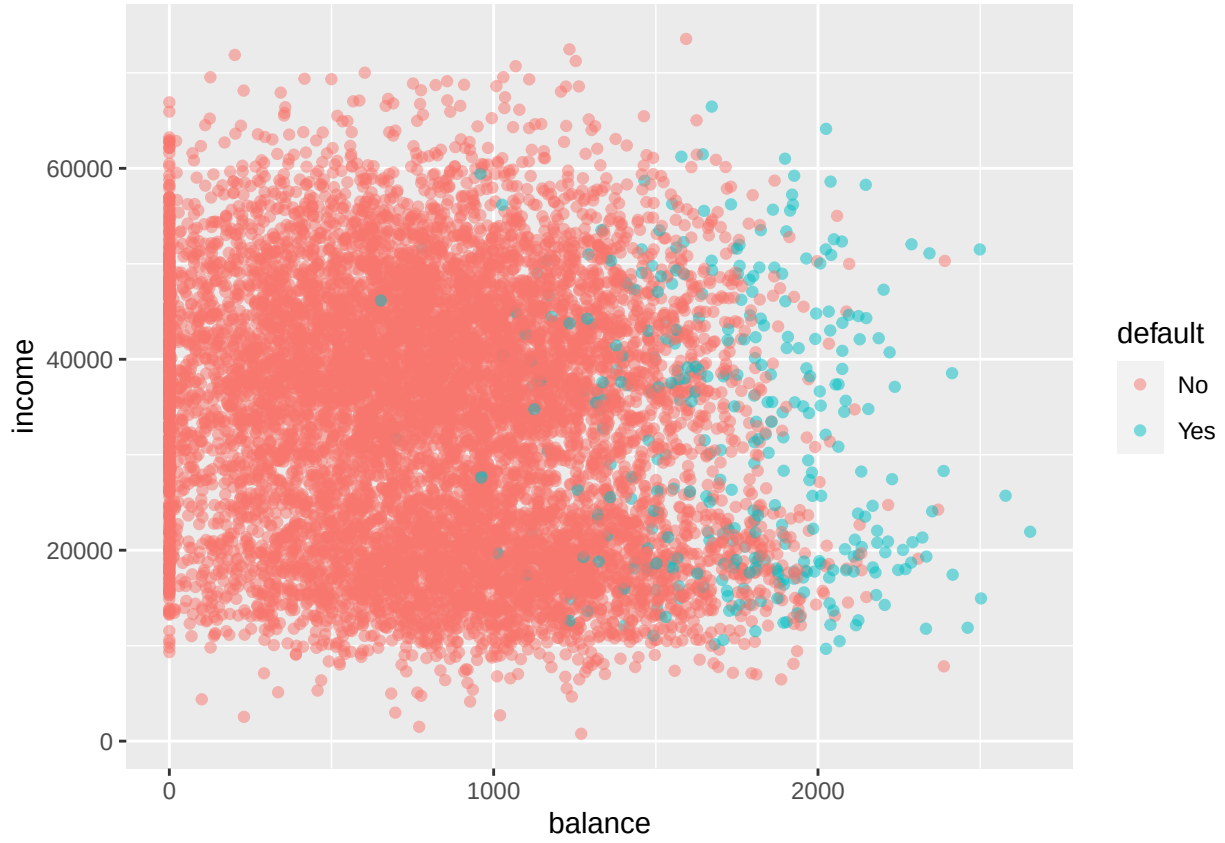
Setting seed

```
set.seed(45)
```

1.

Create a scatterplot of the Default dataset, where balance is mapped to the x position, income is mapped to the y position, and default is mapped to the colour. Can you see any interesting patterns already?

```
Default %>%
  ggplot(aes(x = balance, y= income, col = default)) +
  geom_point(alpha = 0.5)
```

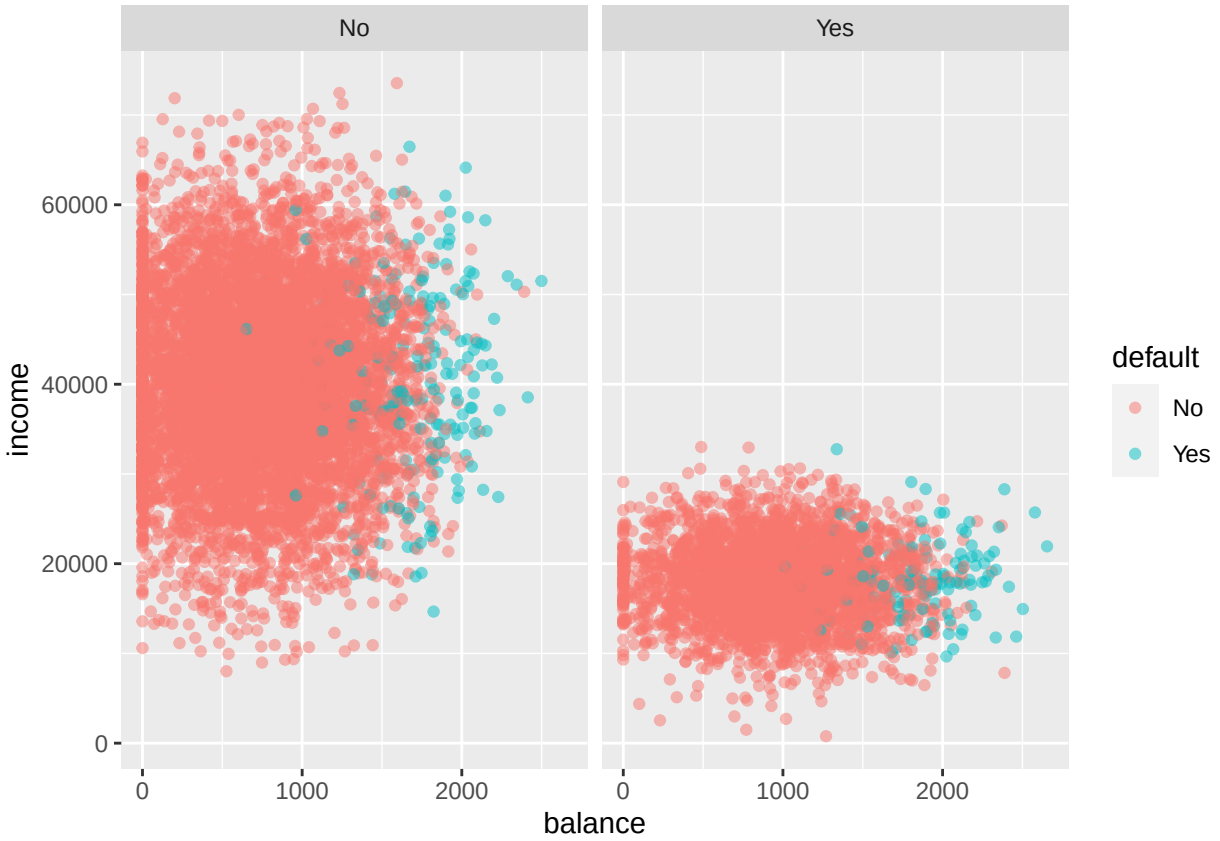


We can already see that people with a higher balance but not necessarily a higher income default their loan. Only three people with a balance lower than 1000 default their loan.

2.

Add facet_grid(cols = vars(student)) to the plot. What do you see?

```
Default %>%
  ggplot(aes(x = balance, y= income, col = default)) +
  geom_point(alpha = 0.5)+
  facet_grid(cols = vars(student))
```



It can be seen that with the faced grid added, the plot is split up by whether or not the person in question is a student or not. The same observations like above can be made. Additionally, almost no student has an income of over 20 000 dollars and in the not-student group only few have an income below 20 000 dollars.

3.

Transform "student" into a dummy variable using ifelse() (0 = not a student, 1 = student). Then, randomly split the Default dataset into a training set default_train (80%) and a test set default_test (20%)

```
Default %<>% mutate(dummy = ifelse(student == "Yes", 1, 0))

train <- sample(1:nrow(Default), round(0.8 * nrow(Default)))
default_train <-  Default[train,]
default_test <- Default[-train,]
```

4.

Create class predictions for the test set using the knn() function. Use student, balance, and income (but no basis functions of those variables) in the default_train dataset. Set k to 5. Store the predictions in a variable called knn_5_pred.

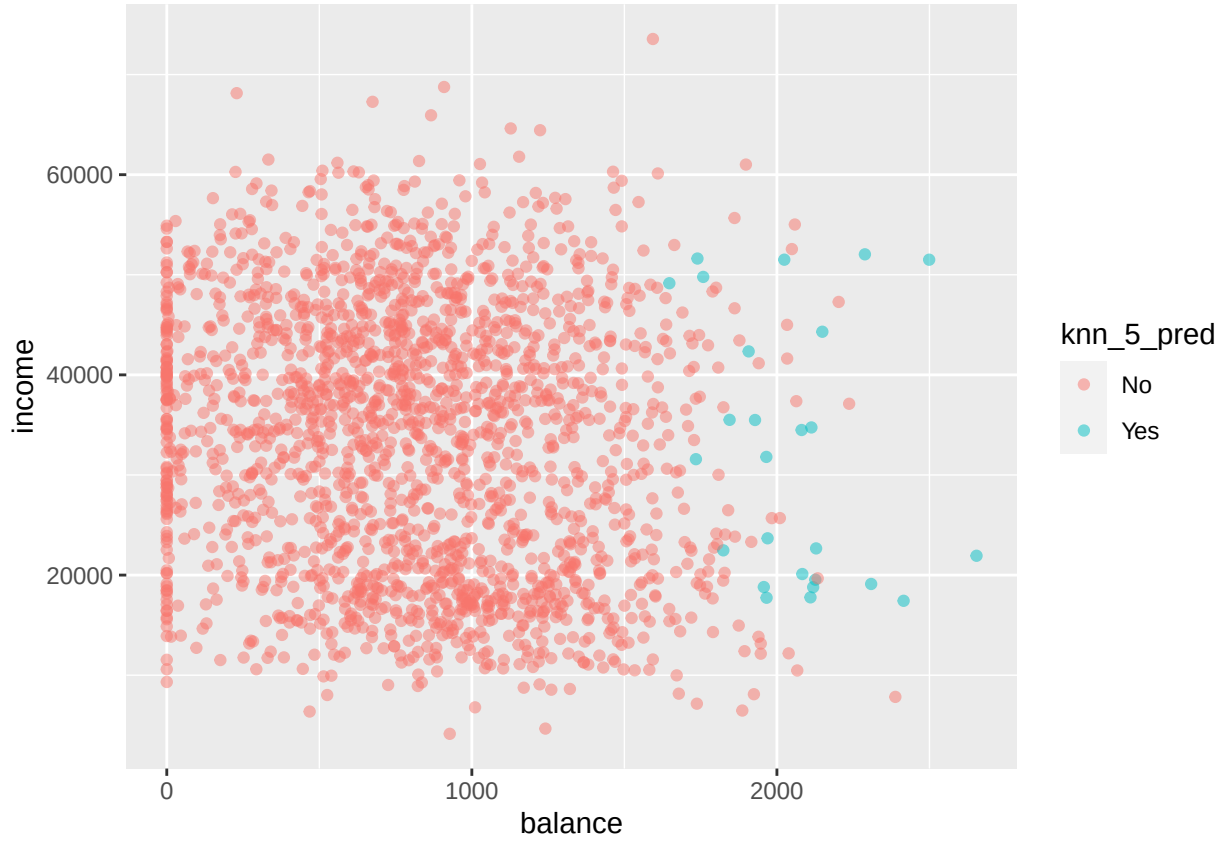
```
knn_5_pred <- knn(default_train[,3:5], default_test[,3:5], #selecting balance, income and dummy (dummy
                  cl = as_factor(default_train$default), k = 5)
```

5.

Create two scatter plots with income and balance as in the first plot you made. One with the true class (default) mapped to the colour aesthetic, and one with the predicted class (knn_5_pred) mapped to the colour aesthetic.

```
default_test %<>% mutate(pred = knn_5_pred)

default_test %>%
  ggplot(aes(x = balance, y= income, col = default)) +
  geom_point(alpha = 0.5)
```

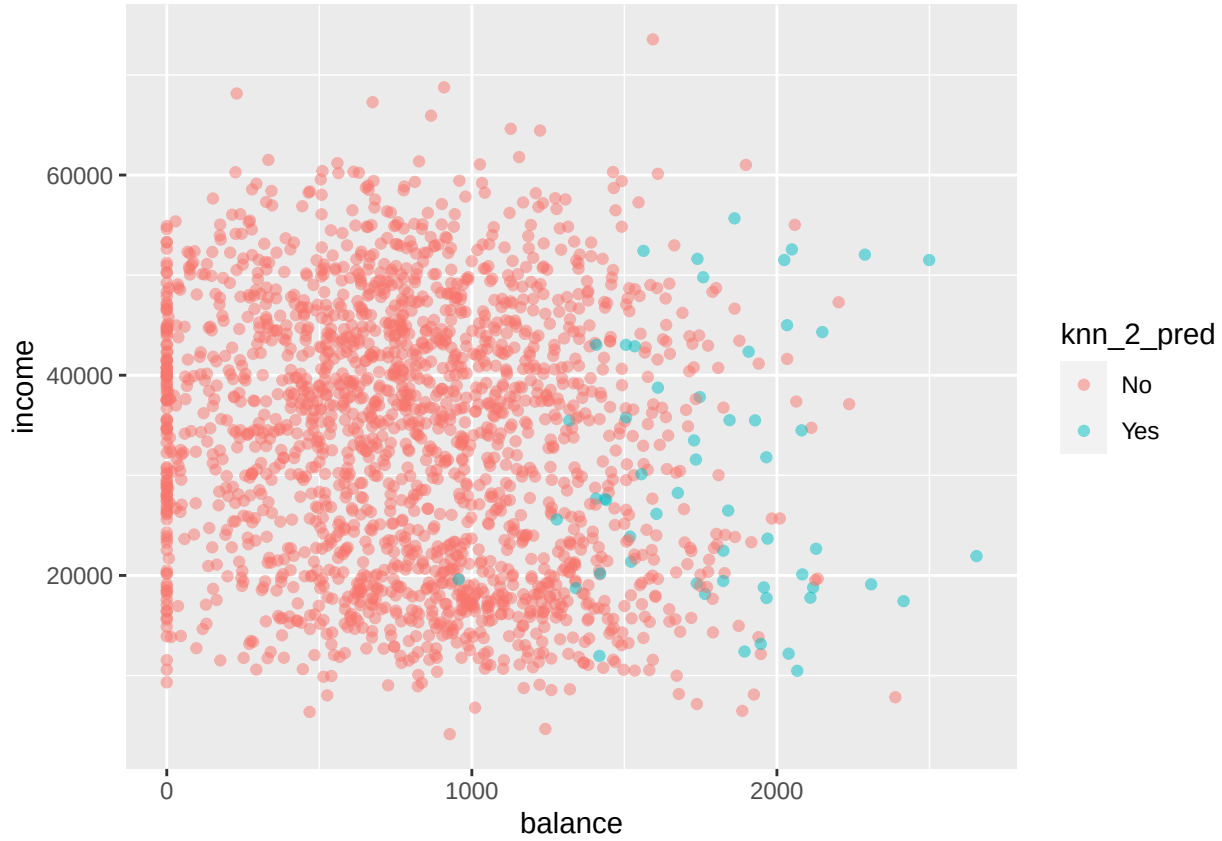
```
default_test %>% ggplot(aes(x = balance, y= income, col = knn_5_pred)) +
  geom_point(alpha = 0.5)
```



6.

Repeat the same steps, but now with a knn_2_pred vector generated from a 2-nearest neighbours algorithm. Are there any differences?

```
knn_2_pred <- knn(default_train[,3:5], default_test[,3:5], #selecting balance, income and dummy (dummy
                  cl = as_factor(default_train$default), k = 2)

default_test %<>% mutate(pred_2 = knn_2_pred)

default_test %>% ggplot(aes(x = balance, y= income, col = knn_2_pred)) +
  geom_point(alpha = 0.5)
```



There are only few differences between k=5 and k=2. Specifically, the cases with high balance and high income are correctly classified as "Yes" for k=2, but "No" for k=5. Also, there are more cases for lower balance that are classified as "Yes" for k=2 which are "No" for k=5. To tell whether those classifications are correct and the right people were classified as "Yes" (k=2) is difficult from just the plots.

7.

What would this confusion matrix look like if the classification were perfect?

```
table(default_test$default)
```

```
##
##   No  Yes
## 1933   67
```

Then we would have 1933 and 67 on the main diagonal of the confusion matrix, meaning that 1933 people were correctly classified as "No" default and 67 correctly classified as "Yes". The other cells would be 0.

8.

Make a confusion matrix for the 5-nn model and compare it to that of the 2-nn model. What do you conclude?

```
table(true = default_test$default, predicted = knn_5_pred)
```

```
##      predicted
## true    No  Yes
##   No  1923   10
##   Yes   51   16
```

```
table(true = default_test$default, predicted = knn_2_pred)
```

```
##      predicted
## true    No  Yes
##   No  1899   34
##   Yes   45   22
```

For 5 nearest neighbor, we get a total of 1939 cases which are correctly classified. 10 people are false false positives, and 51 false negatives.

For 2 nearest neighbor, we get a total of 1914 cases which are correctly classified. 40 people are false false positives, and 46 false negatives.

I conclude that in total, the 5 nearest neighbor procedure performs better classification, especially regarding accuracy.

9.

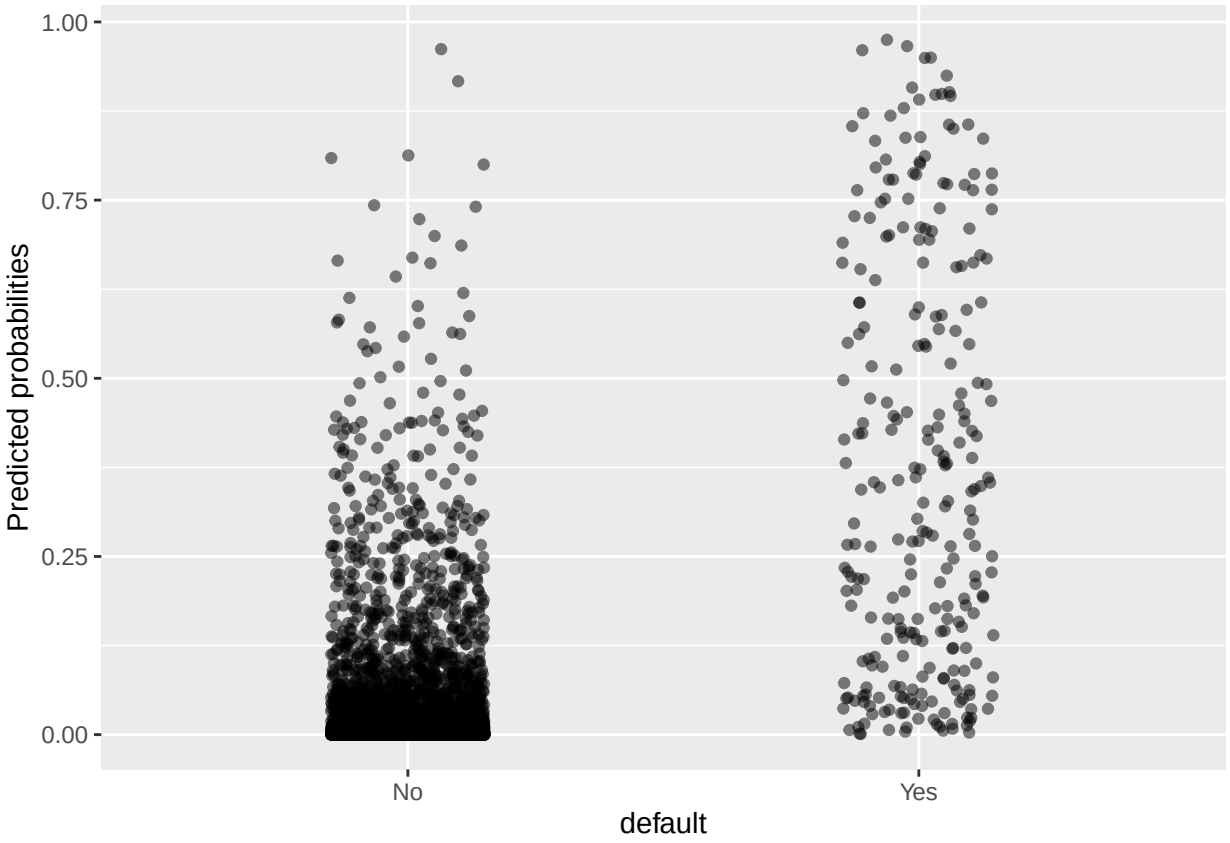
Use glm() with argument family = binomial to fit a logistic regression model lr_mod to the default_train data.

```
lr_mod <- glm(default ~ income + balance + dummy, family = binomial, data = default_train)

#Note that dummy is the dummy version of student.
```

10.

Visualise the predicted probabilities versus observed class for the training dataset in lr_mod. You can choose for yourself which type of visualisation you would like to make. Write down your interpretations along with your plot.

```
lr_pred <- predict(lr_mod, type = "response")

default_train %<>% mutate(pred_lr = lr_pred)
default_train %>% ggplot(aes(x = default, y= pred_lr)) +
  labs(y= "Predicted probabilities")+
  geom_point(position = position_jitter(width = 0.15), alpha = 0.5)
```



It can be seen that individuals with a low predicted probability are more frequently observed as having a "No" on default. This makes sense because we are predicting the probability of category 1 which corresponds to "Yes". However, there are some with low predicted probabilities which are observed to be classified as "Yes" regarding defaulting the loan.

11.

Look at the coefficients of the lr_mod model and interpret the coefficient for balance. What would the probability of default be for a person who is not a student, has an income of 40000, and a balance of 3000 dollars at the end of each month? Is this what you expect based on the plots we've made before?

```
coef(lr_mod)
```

```
##   (Intercept)        income       balance         dummy
## -1.130141e+01  1.338082e-06  6.048276e-03 -6.688593e-01
```

```
exp(coef(lr_mod)[3])
```

```
##  balance
## 1.006067
```

With an increase of 1 in balance for a specific person, the probability of defaulting the loan (when the income stays the same) increases by factor 1.0061.

```
1 / (1 + exp(-(coef(lr_mod)[1] + coef(lr_mod)[2] *40000 + coef(lr_mod)[3] * 3000 +coef(lr_mod)[4] * 0)))
```

```
## (Intercept)
##   0.9989901
```

The probability of default for a person who is not a student, has an income of 40000, and a balance of 3000 dollars at the end of each month is 0.999. This is what we expected from the previous plots as we observed that people with a high income and high balance are likely to have a "Yes" on default (more "Yes" for higher balance and income observed in the plots).

12.

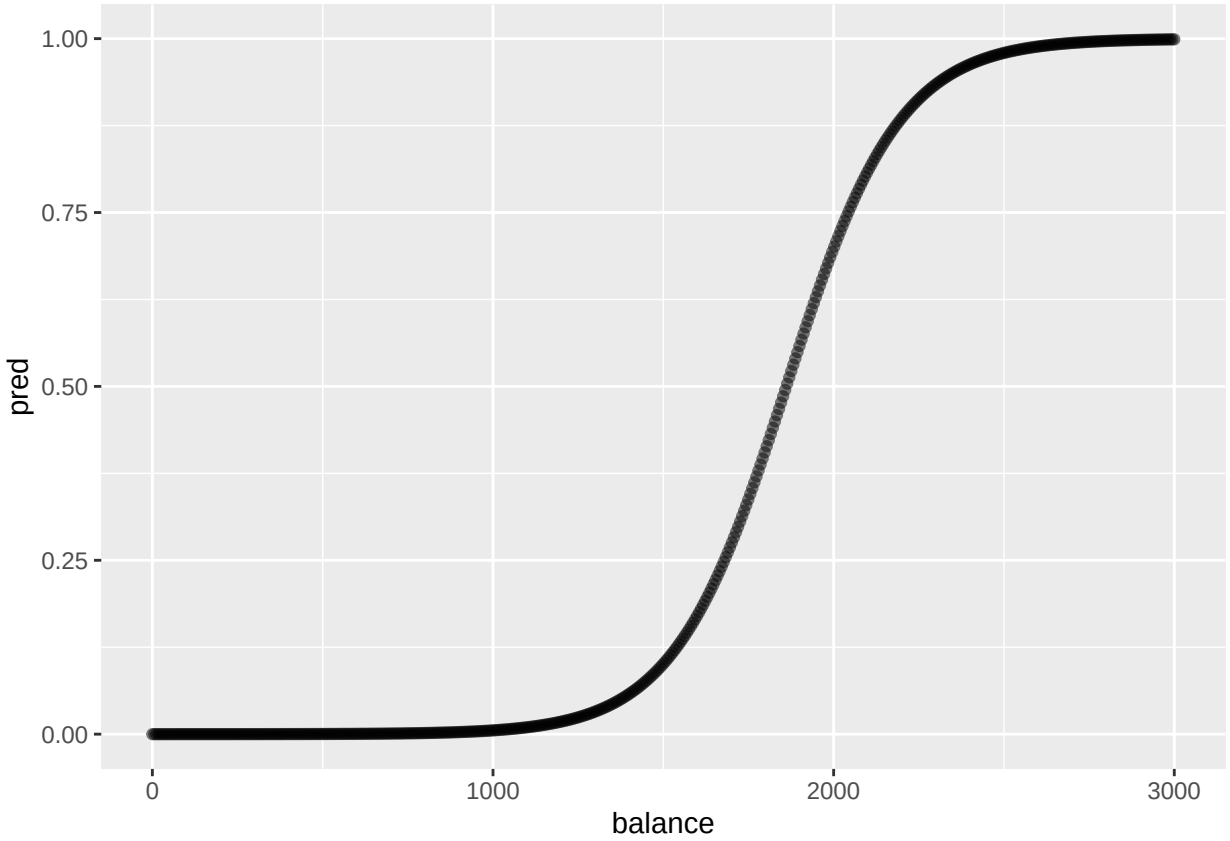
Create a data frame called balance_df with 3 columns and 500 rows: student always 0, balance ranging from 0 to 3000, and income always the mean income in the default_train dataset.

```
balance_df <- data.frame(dummy = rep(0, 500), income = mean(default_train$income), balance= seq(0, 3000
```

13.

Use this dataset as the newdata in a predict() call using lr_mod to output the predicted probabilities for different values of balance. Then create a plot with the balance_df$balance variable mapped to x and the predicted probabilities mapped to y. Is this in line with what you expect?

```
balance_df$pred <- predict(lr_mod, type = "response", newdata = balance_df)

balance_df %>%
  ggplot(aes(x = balance, y= pred)) +
  geom_point(alpha = 0.5)
```



Yes, this is what I expected. For increasing balance > 1000, the probability of defaulting the loan increases if the income stays constant. That it only starts for a balance > 1000 makes sense too, as that is where we start seeing people who default their loan.

14.

Create a confusion matrix just as the one for the KNN models by using a cutoff predicted probability of 0.5. Does logistic regression perform better?

```
pred_lr <- predict(lr_mod, type = "response", newdata = default_test)
pred_lr <- ifelse(pred_lr< 0.5, "No", "Yes")

table(default_test$default, pred_lr)
```

```
##       pred_lr
##          No  Yes
##   No   1921   12
##   Yes    43   24
```

We see that logistic regression performs better than the 2- and 5-nearest neighbor approaches.

15.

Train an LDA classifier lda_mod on the training set.

```
lda_mod <- lda(default ~ income + balance +dummy, default_train)
```

16.

Look at the lda_mod object. What can you conclude about the characteristics of the people who default on their loans?

```
lda_mod
```

```
## Call:
## lda(default ~ income + balance + dummy, data = default_train)
##
## Prior probabilities of groups:
##      No     Yes
## 0.96675 0.03325
##
## Group means:
##       income   balance     dummy
## No  33577.25  805.3601 0.2922162
## Yes 31668.13 1757.2081 0.3984962
##
## Coefficients of linear discriminants:
##                    LD1
## income   3.293756e-06
## balance  2.245908e-03
## dummy   -1.397098e-01
```

People who default their loans are more likely to be students, have a higher balance and lower income than people who do not default their loans.

17.

Create a confusion matrix and compare it to the previous methods.

```
pred_lda <- predict(lda_mod, newdata = default_test)

table(default_test$default, pred_lda$class)
```

```
##
##         No  Yes
##   No  1925    8
##   Yes   51   16
```

The total of correctly classified people is 1941. Hence, this method has a slightly worse accuracy than the logistic regression (1945). But the accuracy is higher than for the k-nearest neighbors approaches.

Compared to the logistic regression approach, there are (8) more false positives and (4) less false negatives when using the linear discriminant analysis.

18.

Create a model (using knn, logistic regression, or LDA) to predict whether a 14 year old boy from the 3rd class would have survived the Titanic disaster. You can find the data in the data/ folder. Would the passenger have survived if they were a girl in 2nd class?

```
ourTitanic <- read.csv("Titanic.csv")

#logistic regression model
lr_titan <- glm(Survived ~ Age + PClass + Sex, family= binomial, data = ourTitanic)

lr_titan
```

```
##
## Call:  glm(formula = Survived ~ Age + PClass + Sex, family = binomial,
##     data = ourTitanic)
##
## Coefficients:
## (Intercept)          Age    PClass2nd    PClass3rd      Sexmale
##     3.75966     -0.03918     -1.29196     -2.52142     -2.63136
##
## Degrees of Freedom: 755 Total (i.e. Null);  751 Residual
##   (557 Beobachtungen als fehlend gelöscht)
## Null Deviance:       1026
## Residual Deviance: 695.1     AIC: 705.1
```

```
#Seeing that female and first class are the reference categories.

# probability
prob_boy <- 1 / (1 + exp(-(coef(lr_titan)[1] + 14 *coef(lr_titan)[2] + 0 * coef(lr_titan)[3]+ 1 *coef(li

prob_boy
```

```
## (Intercept)
##   0.1254734
```

```
prob_girl <- 1 / (1 + exp(-(coef(lr_titan)[1] + 14 *coef(lr_titan)[2] + 1 * coef(lr_titan)[3]+ 0 *coef(l

prob_girl
```

```
## (Intercept)
##   0.8720519
```

The 14 year-old boy in class three only has a survival probability of 12.55%. Thus, we would classify him as not surviving. On the other hand, the 14 year-old girl in second class is likely to survive (87.21%) and we would classify her as surviving.