KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

Vytautas Kraujalis

6 variantas

Studijų modulio

P160M101 DAUGIAMATĖ STATISTINĖS ANALIZĖ

2 laboratorinis darbas

KAUNAS, 2021

UŽDUOTIS

MNIST duomenų rinkinyje yra surinkti ranka rašytų skaitmenų vaizdai. Kadangi vaizdų dydis yra 28x28 taškų, tai kiekvienas vaizdas apibūdinamas 784 požymių. Naudodami šį duomenų rinkinį atlikite šias užduotis:

1) Susidarykite savo individualią imtį į ją įtraukdami stebėjimus atitinkančius šiuos skaitmenis:
   1 var.: 0123
   2 var.: 2345
   3 var.: 4567
   4 var.: 6789
   5 var.: 0246
   6 var.: 1357
   7 var.: 0123 Fashion MNIST duomenų rinkiniui
   8 var.: 2345 Fashion MNIST duomenų rinkiniui
   9 var.: 4567 Fashion MNIST duomenų rinkiniui
   10 var.: 6789 Fashion MNIST duomenų rinkiniui
   11 var.: 0246 Fashion MNIST duomenų rinkiniui
   12 var.: 1357 Fashion MNIST duomenų rinkiniui
   13 var.: 2570 Fashion MNIST duomenų rinkiniui
2) Pavaizduokite po kelis kiekvienos klasės objektus (vaizdus).
3) Panaudokite pagrindinių komponenčių metodą. Pavaizduokite grafiką vaizduojantį pirmų 20 pagrindinių komponenčių dispersijas.
4) Kokią suminės dispersijos dalį savyje turi pirmos 4 pagrindinės komponentės? Taip pat apskaičiuokite, kelių pagrindinių komponenčių pakanka, kad turėti 90% suminės dispersijos?
5) Pavaizduokite pirmų dviejų komponenčių sklaidos diagramą. Diagramoje esančius taškus vaizduokite skirtingomis spalvomis priklausomai nuo skaitmens, kurį jie atitinka.
6) Pavaizduokite analogiškas dvimates sklaidos diagramas trims atsitiktinėms pradinių požymių poroms. Palyginkite galimybes klasifikuoti ranka rašytus skaitmenis naudodami atsitiktines koordinates ir koordinates gautas naudojant pagrindinių komponenčių metodą.

Atrinkite iš duomenų imties tik stebėjimus, kur klasė *label* yra lygi vienai iš pirmų dviejų reikšmių nurodytų prie grupės varianto. Tolimesnius užduoties punktus atlikite tik su šia imtimi turinčia dvi klases.

7) Sukurkite dvinarės logistinės regresijos modelius skaitmens prognozei pagal vieną, dvi, ..., dvidešimt pirmųjų pagrindinių komponenčių. Rezultatą palyginkite su pažingsniniu pirmyn (forward) metodu, kuris įtraukia vieną, dvi ir t.t. reikšmingiausias pagrindines komponentes. Nubraižykite grafiką vaizduojantį prognozių tikslumo priklausomybes nuo modelių regresorių kiekio.
8) Naudodami pažingsninį pirmyn (forward) regresijos kintamųjų atrinkimo metodą, sudarykite dvinarės logistinės regresijos modelius su vienu, dviem, ..., dvidešimt (ar mažiau, jei pažingsnisnis metodas daugiau kintamųjų neįtraukė) pagrindinių komponenčių ta eilės tvarka, kaip is pažingsniniame metode. Ar pagrindinės komponentės įtraukiamos į modelį jų eilės tvarka?
9) Išrinkite 20 atsitiktinių požymių (pvz. naudodami funkciją runif(), sample() ar pan.). Sukurkite dvinarės logistinės regresijos modelius skaitmens prognozei pagal vieną, du, ..., dvidešimt atsitiktinių požymių iš anksčiau sudaryto 20 atsitiktinių požymių sąrašo. Nubraižykite grafiką vaizduojantį prognozės tikslumo priklausomybė nuo modeliui pasirinktų požymių kiekio.
10) Naudodami pažingsninį pirmyn (forward) regresijos kintamųjų atrinkimo metodą, sudarykite dvinarės logistinės regresijos modelius su vienu, dviem, ..., dvidešimt (ar mažiau, jei pažingsnisnis metodas daugiau kintamųjų neįtraukė) visų pradinių kintamųjų ta eilės tvarka, kaip is pažingsniniame metode. Apskaičiuokite šių metodų tikslumą, pateikite tikslumo priklausomybės nuo komponenčių kiekio grafiką.
11) Nubraižykite 7, 8, 9 ir 10 užduoties punkto grafikus viename paveikslėlyje. Padarykite išvadas apie modelių tikslumą ir kintamųjų ar pagrindinių komponenčių infomratyvumą klasifikavime.
12) Atliksime skaičiavimus naudodami dvinarės logistinės regresijos modelį su L1 ir L2 reguliarizavimais. Atvaizduokite koeficientų priklausomybės nuo reguliarizavimo konstantos λ kitimo grafiką kiekvienai iš reguliarizacijų. Palyginkite atvejus, kai į modelį įtraukiami:
    a. visi pradiniai požymiai;
    b. kai įtraukiamos pagrindinės komponentės.

    Jeigu susidursite su skaičiavimo laiko problema, galite mažinti pradinių kintamųjų ir pagrindinių komponenčių kiekį.

13) Parinkite L1 modelio reguliarizavimo konstantą 12 punkto užduotyse taip, kad į modelį būtų įtraukta a) 10 pradinių požymių; b) 10 pagrindinių komponenčių. Aprašykite gautus rezultatus: kurios pagrindinės komponentės (kurie požymiai) buvo įtrauktos į modelį? Rezultatus palyginkite su pažingsninio metodo rezultatais.

Užduoties ataskaita pateikiama per http://moodle.ktu.lt/ Ataskaitoje reikia pateikti visas užduotis atitinkantį pilną R kodą, bei jo įvykdymo rezultatą (tekstinę ar grafinę), atsakyti į užduotyse nurodytus klausimus, aprašyti sunkumus su kuriais susidūrėte ir jų sprendimo būdus.

SPRENDIMAS

1 užduotis

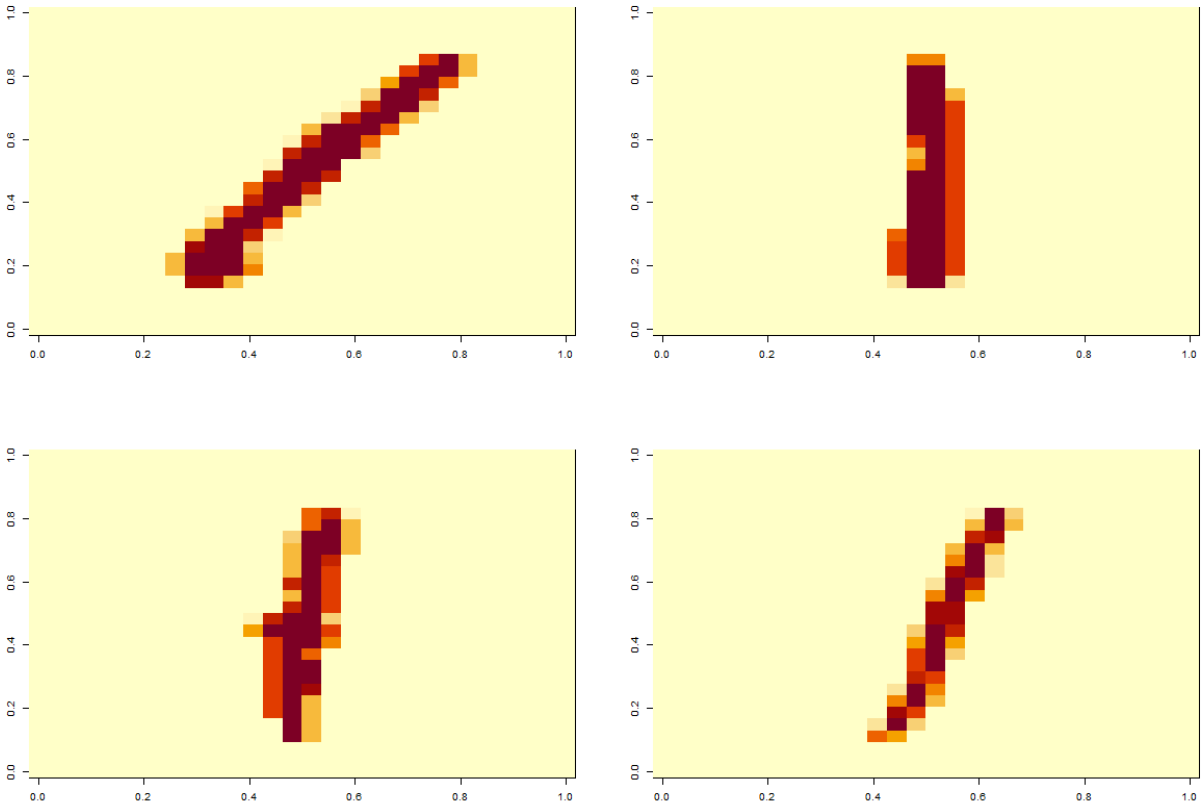
```
library(dplyr)
library(factoextra)
library(caret)
library(glmnet)

data <- read.table("mnist.txt", sep = ",", header = T) %>%
   filter(label %in% c(1, 3, 5, 7)) # 6 var.: 1357
```

6 variantas, todėl pasirinkti stebėjimai, atitinkantys 1, 3, 5, 7 skaitmenis.

2 užduotis

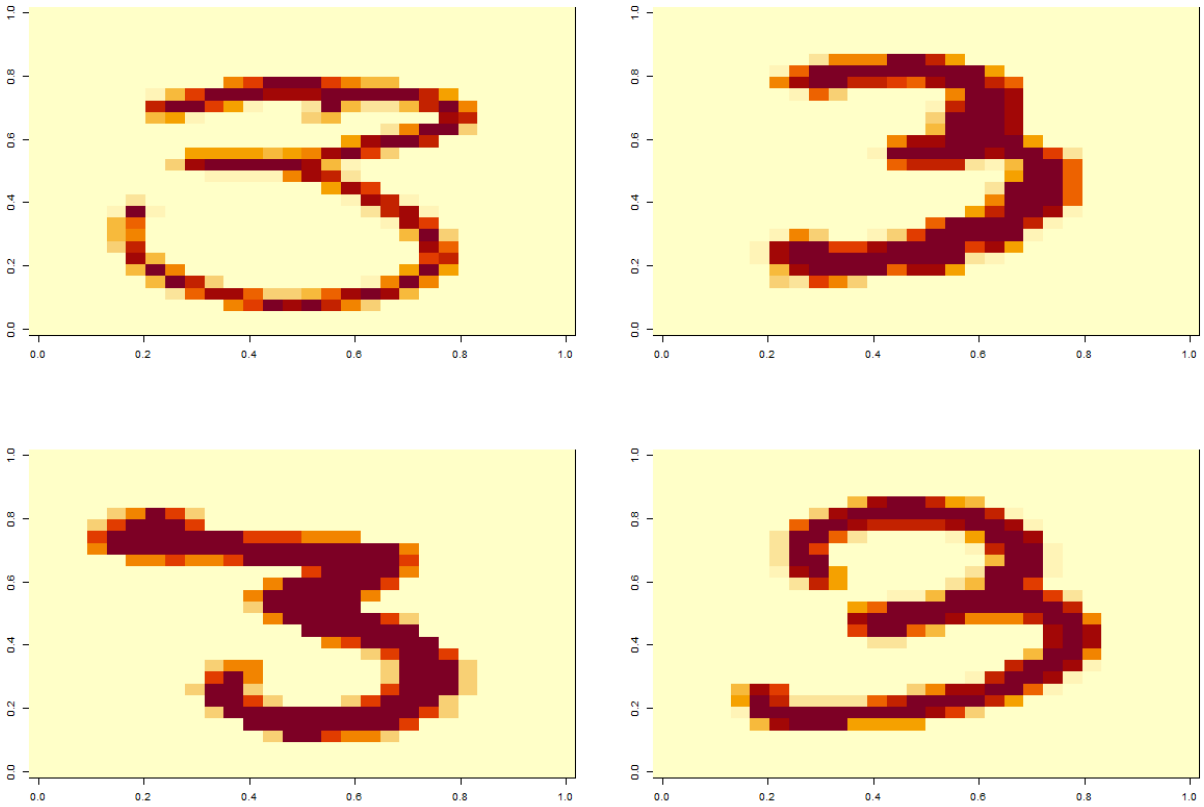
Vienetų grafikas:

```
spausdinti_grafika <- function(skaicius, kuri_irasa){
   image(t(matrix(data %>% filter(label == skaicius) %>% slice(kuri_irasa) %>
% select(-label) %>% t() %>% as.vector(), 28, byrow = T))[1:28, 28:1])
}

# vienetai:
png(file = "vienetai.png", width = 1200, height = 850)
par(mfrow=c(2,2))
spausdinti_grafika(1, 1)
spausdinti_grafika(1, 2)
spausdinti_grafika(1, 3)
spausdinti_grafika(1, 4)
dev.off()
```



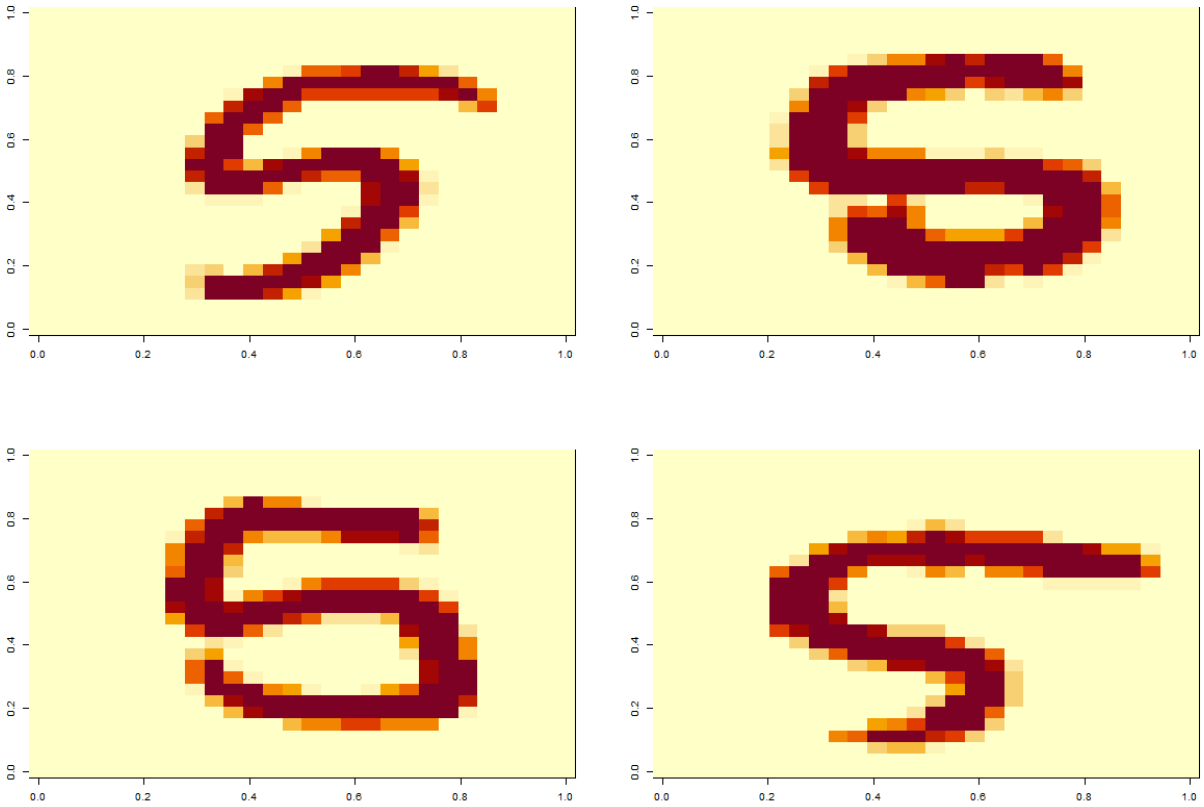
Trejetų grafikas:

```
# trejetai:
png(file = "trejetai.png", width = 1200, height = 850)
par(mfrow=c(2,2))
spausdinti_grafika(3, 1)
spausdinti_grafika(3, 2)
spausdinti_grafika(3, 3)
spausdinti_grafika(3, 4)
dev.off()
```



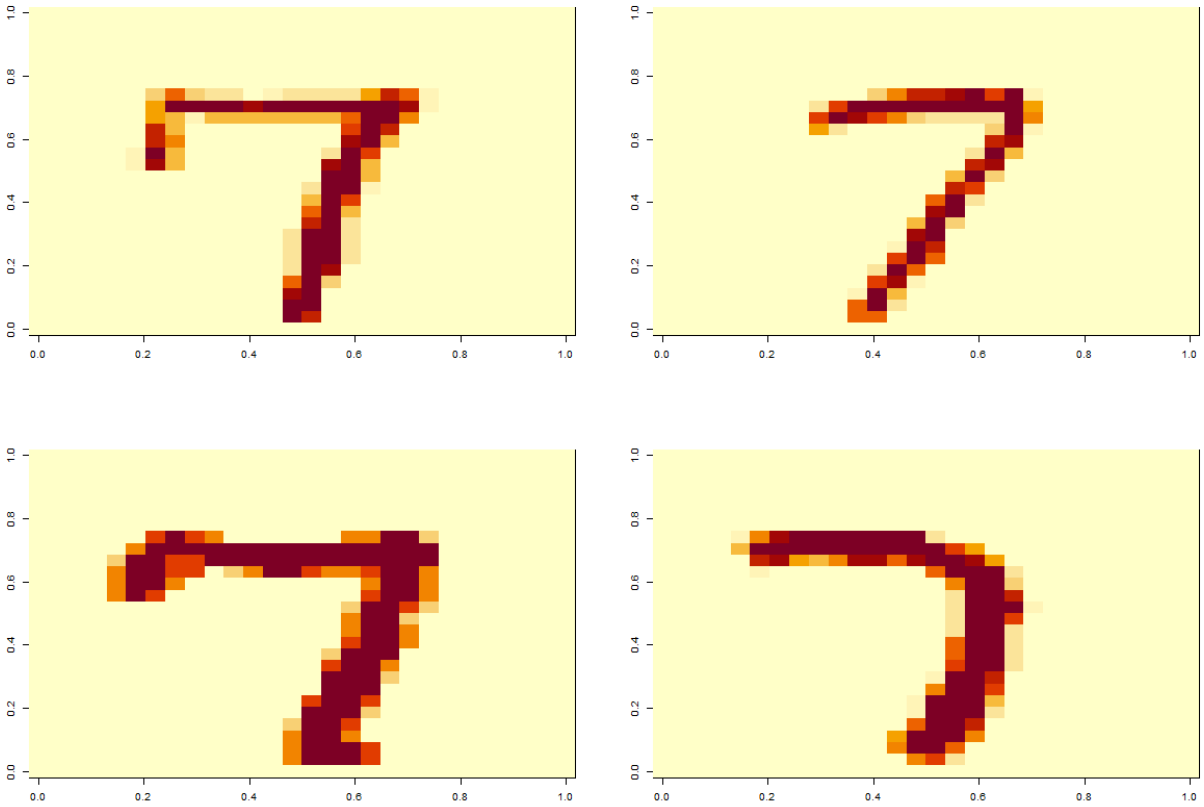
Penketų grafikas:

```
# penketai:
png(file = "penketai.png", width = 1200, height = 850)
par(mfrow=c(2,2))
spausdinti_grafika(5, 1)
spausdinti_grafika(5, 2)
spausdinti_grafika(5, 3)
spausdinti_grafika(5, 4)
dev.off()
```



Septynetų grafikas:

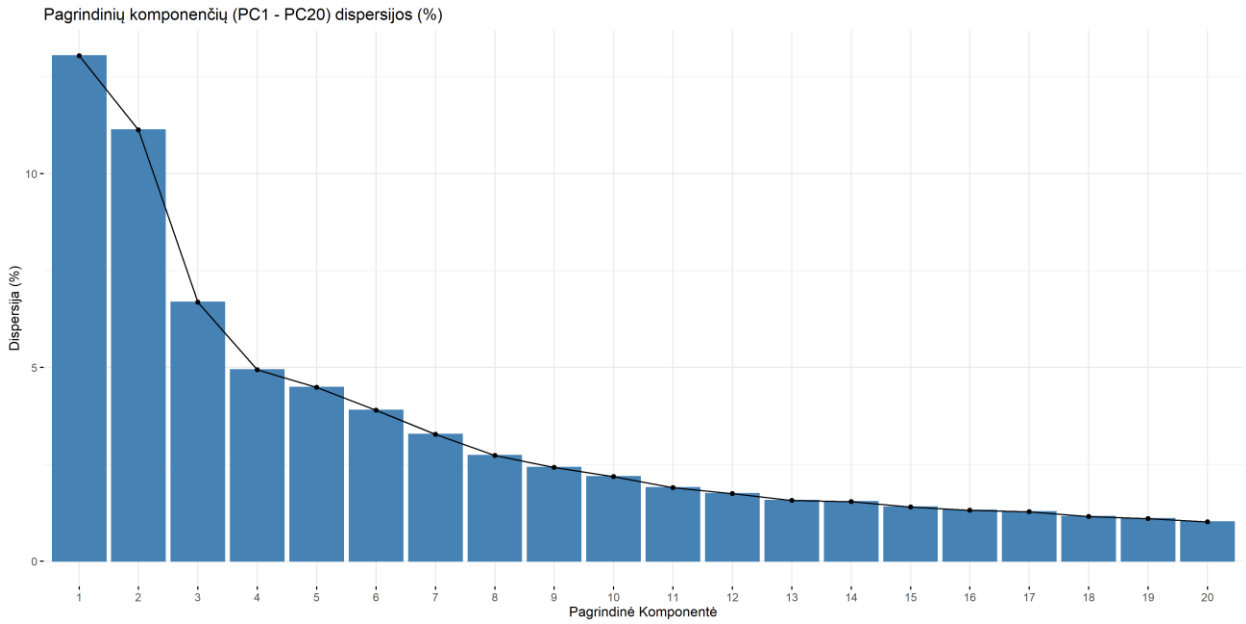
```
# septynetai:
png(file = "septynetai.png", width = 1200, height = 850)
par(mfrow=c(2,2))
spausdinti_grafika(7, 1)
spausdinti_grafika(7, 2)
spausdinti_grafika(7, 3)
spausdinti_grafika(7, 4)
dev.off()
```



Matom, kad tarp tų pačių skaitmenų gali būti įvarių užrašymo būdų.

3 užduotis

```
pca <- prcomp(data %>% select(-label), scale = FALSE)
fviz_eig(pca, ncp = 20, main = "Pagrindinių komponenčių (PC1 - PC20) dispersi
jos (%)", ylab = "Dispersija (%)", xlab = "Pagrindinė Komponentė")

ggsave(filename = "pca_dispersijos.png", width = 14, height = 7, units = "in"
, bg = "white")
```



Pateikta % išraiška, kiek kuri komponentė paaiškina dispersijos, atvaizduotos pirmos 20 pagrindinės komponentės.

4 užduotis

```
pca_var <- pca$sdev^2
pca_var <- pca_var / sum(pca_var)

print(paste0("Pirmos 4 pagrindinės komponentės sumiškai savyje turi ", round(
sum(pca_var[1:4]) * 100, digits = 2), "% dispersijos."))
```

```
## [1] "Pirmos 4 pagrindinės komponentės sumiškai savyje turi 35.79% dispersi
jos."
```

Pirmos 4 pagrindinės komponentės sumiškai savyje turi 35.79% dispersijos.

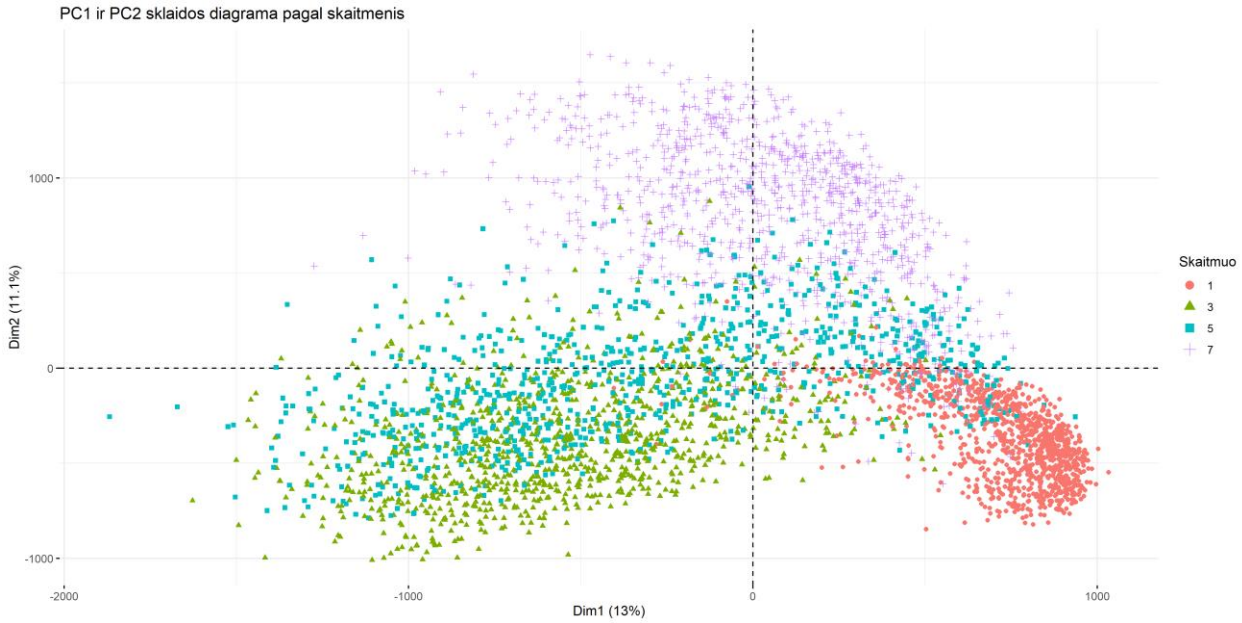
```
dispersija <- 0
kiek_pca <- 0
while(dispersija < 0.9){
   kiek_pca <- kiek_pca + 1
   dispersija <- sum(pca_var[1:kiek_pca])
}

print(paste0("Tam, kad pasiektume 90% suminės dispersijos, mums reikia panaud
oti pirmas ", kiek_pca, " pagrindines komponentes, su šiomis komponentinėmis,
suminė dispersija sieka ", round(dispersija * 100, digits = 2), "%"))
```

```
## [1] "Tam, kad pasiektume 90% suminės dispersijos, mums reikia panaudoti pi
rmas 78 pagrindines komponentes, su šiomis komponentinėmis, suminė dispersija
sieka 90.08%"
```

Tam, kad pasiektume 90% suminės dispersijos, mums reikia panaudoti pirmas 78 pagrindines komponentes, su šiomis komponentinėmis, suminė dispersija sieka 90.08%

5 užduotis

```
skaitmenys <- as.factor(data$label)
fviz_pca_ind(pca,
             col.ind = skaitmenys, # color by groups
             legend.title = "Skaitmuo",
             label = "none",
             title = "PC1 ir PC2 sklaidos diagrama pagal skaitmenis"
             )

ggsave(filename = "PC_1_2_sklaida.png", width = 14, height = 7, units = "in",
bg = "white")
```



Atvaizdavus pirmas 2 pagrindines komponentes sklaidos diagrama, galima nesunkiai klasifikuoti 1 ir 7 skaitmenis. 3 ir 5 skaitmenys yra kiek sunkiai klasifikuojami tarpusavyje, kadangi turi bendra „regioną“ kur abi klasės persipina.

6 užduotis

```
uzpildyti_pixeliai <- data %>%
   sapply(function(x){sum(x > 0)}) %>%
   sort(decreasing = TRUE)

uzpildyti_pixeliai <- uzpildyti_pixeliai[uzpildyti_pixeliai / nrow(data) > 0.
1]
uzpildyti_pixeliai <- uzpildyti_pixeliai[-1]
```

Kadangi yra stulpelių (pixelių), kurie jokiose eilutėse nebūna užpildyti, atfiltruosim tik tuos stulpelius, kurie užpildyti (reikšmė > 0) bent 10% iš stebėjimų. Iš gautų stulpelių atrinksime atsitiktines 3 poras ir atvaizduosime grafiškai.

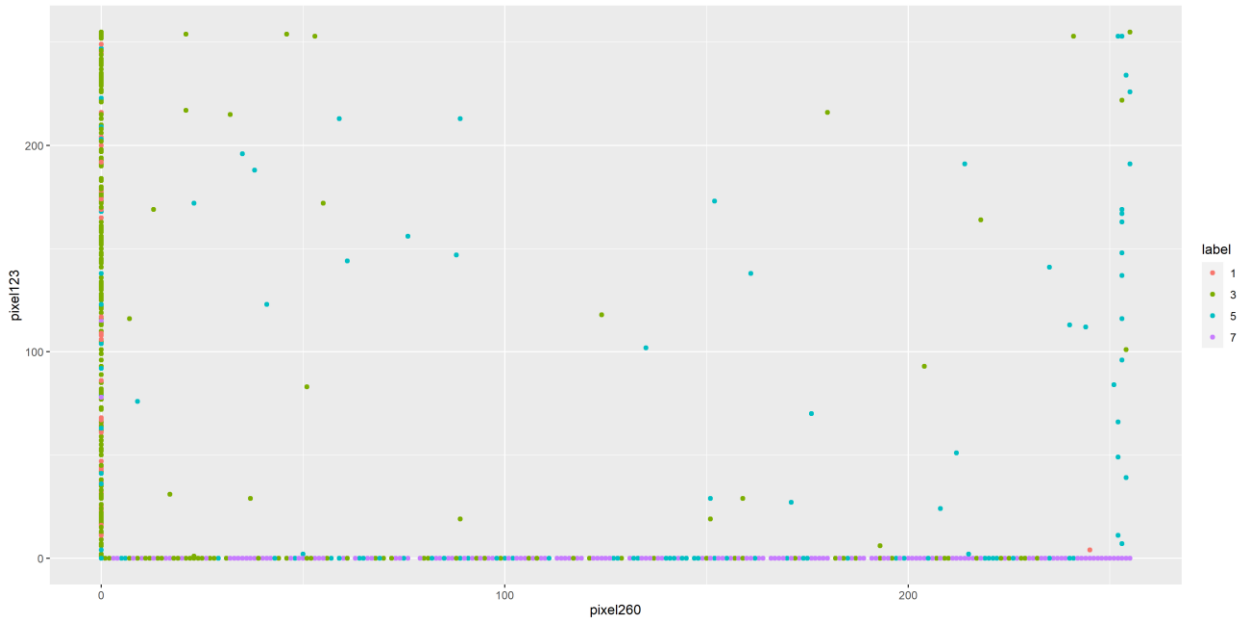
Atrenkam 6 atsitiktinius stulpelius:

```
atsitiktines_poros <- sample(names(uzpildyti_pixeliai), 6, replace = F)
```

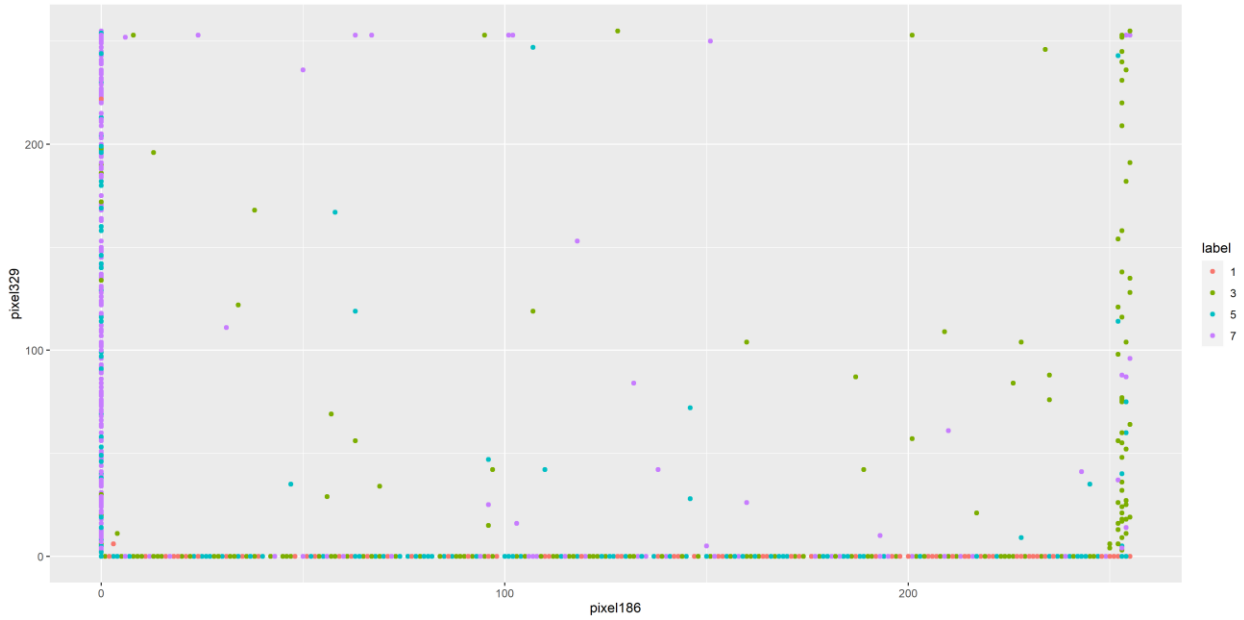
Atvaizduojame 3 atsitiktines poras:

```
atsitiktiniai_data <- data %>%
   select(all_of(atsitiktines_poros), label) %>%
   mutate(label = as.factor(label))
atsitiktiniai_data_colnames <- colnames(atsitiktiniai_data)
colnames(atsitiktiniai_data) <- c("V1", "V2", "V3", "V4", "V5", "V6", "label"
)

atsitiktiniai_data %>%
   ggplot(aes(x = V1, y = V2, color = label)) +
   geom_point() +
   xlab(atsitiktiniai_data_colnames[1]) +
   ylab(atsitiktiniai_data_colnames[2])

ggsave(filename = "atsitiktines_sklaidos_1.png", width = 14, height = 7, unit
s = "in", bg = "white")
```


```
atsitiktiniai_data %>%
   ggplot(aes(x = V3, y = V4, color = label)) +
   geom_point() +
   xlab(atsitiktiniai_data_colnames[3]) +
   ylab(atsitiktiniai_data_colnames[4])

ggsave(filename = "atsitiktines_sklaidos_2.png", width = 14, height = 7, unit
s = "in", bg = "white")
```




```
atsitiktiniai_data %>%
   ggplot(aes(x = V5, y = V6, color = label)) +
   geom_point() +
   xlab(atsitiktiniai_data_colnames[5]) +
   ylab(atsitiktiniai_data_colnames[6])

ggsave(filename = "atsitiktines_sklaidos_3.png", width = 14, height = 7, unit
s = "in", bg = "white")
```

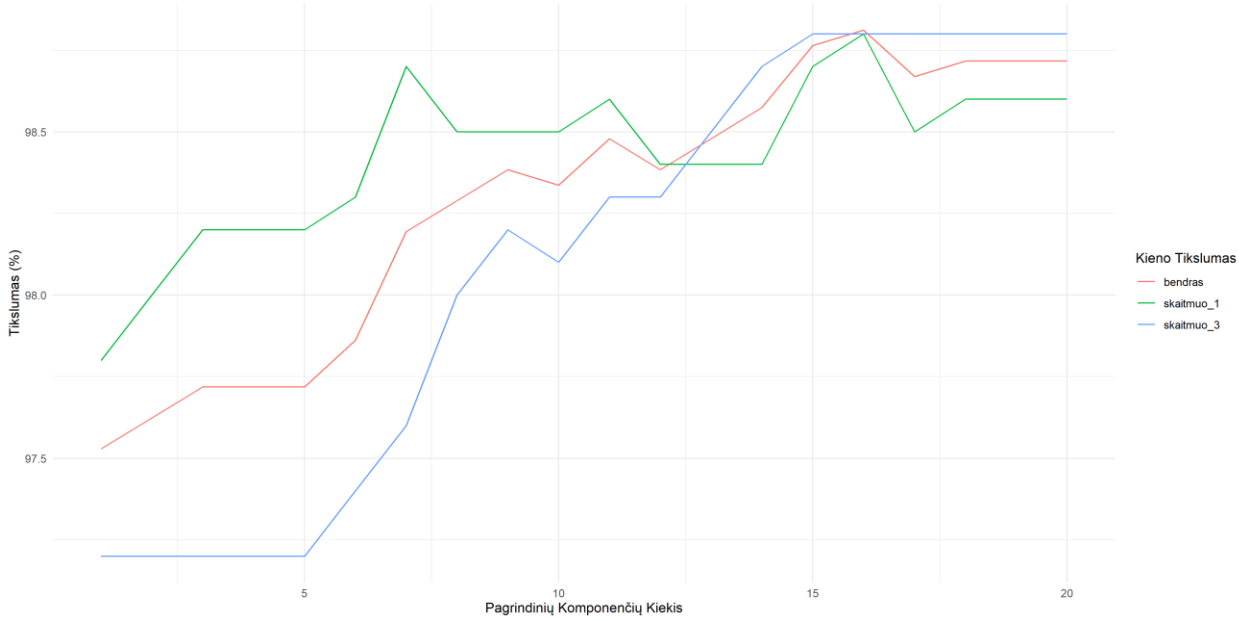


Akivaizdžiai matosi, jog naudojat 2 atsitiktinius požymius (pikselius), klasifikuoti skaitments praktiškai neįmanoma, kadangi klasės tarpusavyje yra stipriai susimaišiusios. Tačiau vaizduojant pirmas 2 pagrindine komponentes, galima būtų klasifikuoti gana aukštu tikslumu, 1 ir 7 skaitmenys gana gerai atsiskiria nuo kitų klasių.

7 užduotis

```
data <- data %>%
   filter(label %in% c(1, 3)) %>%
   mutate(label = as.factor(label))

pca <- prcomp(data %>% select(-label), scale = FALSE)

data_pcs <- pca$x %>%
   bind_cols(label = data$label)

# 2 klasių tikslumo ištraukimui iš sumaišymo matricos
classAcc <- function(confusionMatrix) {
  class1 <- round(confusionMatrix$table[1, 1] / sum(confusionMatrix$table[, 1
]) * 100, 1)
  class2 <- round(confusionMatrix$table[2, 2] / sum(confusionMatrix$table[, 2
]) * 100, 1)
  bendras <- confusionMatrix$overall["Accuracy"] * 100
  acc <- c(class1, class2, bendras)
  names(acc) <- c(colnames(confusionMatrix$table), "Bendras")
  return(acc)
}

# Susidarom list'ą su 20 formulių
formules <- NULL
formules[[1]] <- "label ~ PC1"
for(i in 2:20){
   formules[[i]] <- "label ~ PC1"
   for(j in 2:i){
      formules[[i]] <- paste0(formules[[i]], " + PC", j)
   }
}

formules <- lapply(formules, as.formula)

tikslumai <- NULL
for(i in 1:20){
   glm <- glm(formula = formules[[i]], data = data_pcs, family = "binomial")
   prob_pred = predict(glm, type = "response", newdata = data_pcs %>% select(
-label))
   y_pred = ifelse(prob_pred > 0.5, 3, 1) %>%
      as.factor()
   confusion_matrix <- confusionMatrix(y_pred, data_pcs$label)
   tikslumai[[i]] <- classAcc(confusion_matrix)
}


tikslumai_data <- do.call(rbind.data.frame, tikslumai) %>%
   bind_cols(kiek_pca = 1:20)
colnames(tikslumai_data) <- c("skaitmuo_1", "skaitmuo_3", "bendras", "kiek_pc
a")


tikslumai_data %>%
```

```
   tidyr::pivot_longer(c(-kiek_pca), names_to = "Kieno Tikslumas", values_to
= "Tikslumas") %>%
   ggplot(aes(x = kiek_pca, y = Tikslumas, color = `Kieno Tikslumas`)) +
   geom_line() +
   theme_minimal() +
   xlab("Pagrindinių Komponenčių Kiekis") +
   ylab("Tikslumas (%)")

ggsave(filename = "tikslumas_pca.png", width = 14, height = 7, units = "in",
bg = "white")
```



Logistinės regresijos modelis naudojant 1 – 20 pagrindinių komponenčių. Pagal tikslumą matom, jog didžiausias tikslumas pasiekiamas naudojant ne visas pagrindines komponentes, tačiau skirtumai yra labai minimalūs.

```
glm <- glm(label ~ ., data = data_pcs[,c(1:20, 785)], family = "binomial")
glm_tuscias <- glm(label ~ 1, data = data_pcs, family = "binomial")
# Forward
forward_glm <- step(glm_tuscias, direction = "forward", scope = formula(glm),
trace=0)
summary(forward_glm)

##
## Call:
## glm(formula = label ~ PC1 + PC6 + PC7 + PC8 + PC16 + PC11 + PC13 +
##     PC15 + PC12 + PC20 + PC9 + PC5 + PC10 + PC4 + PC14, family = "binomial
",
##     data = data_pcs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.5705  -0.0384  -0.0098   0.0058   1.9808
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
```

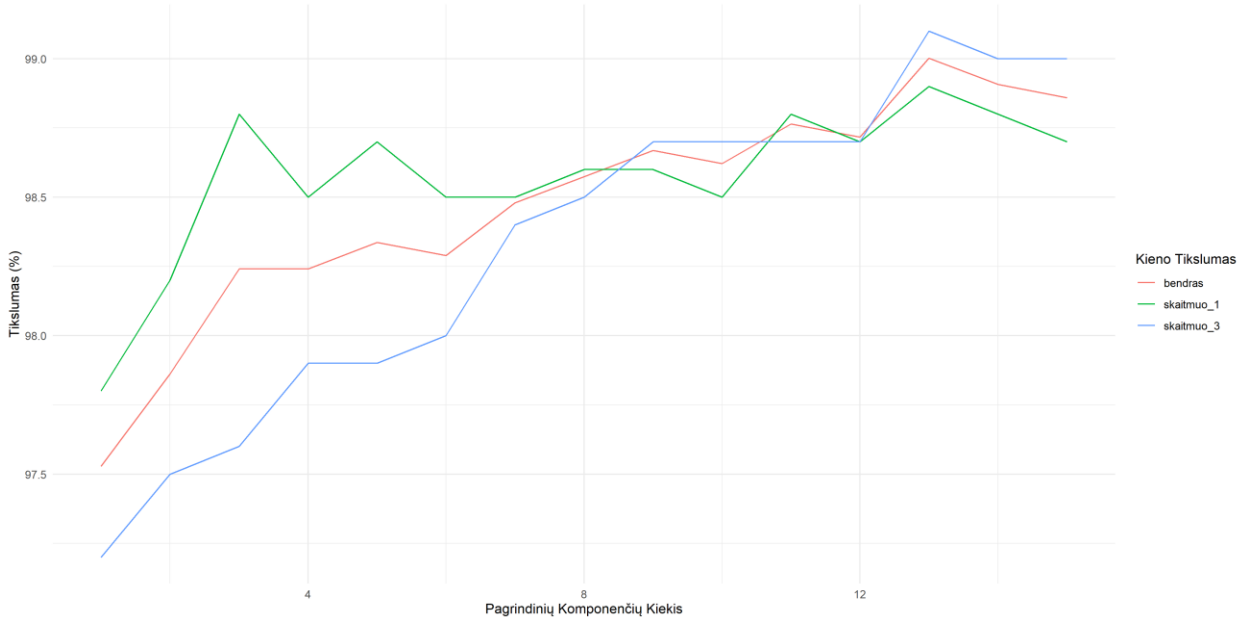
```
## (Intercept)  1.4421196  0.4398288   3.279 0.001042 **
## PC1         -0.0124006  0.0012118 -10.233  < 2e-16 ***
## PC6          0.0050536  0.0009631   5.247 1.54e-07 ***
## PC7         -0.0043165  0.0010998  -3.925 8.67e-05 ***
## PC8         -0.0039978  0.0010385  -3.850 0.000118 ***
## PC16         0.0068052  0.0016223   4.195 2.73e-05 ***
## PC11         0.0037038  0.0010448   3.545 0.000392 ***
## PC13        -0.0022650  0.0011576  -1.957 0.050400 .
## PC15        -0.0034864  0.0013118  -2.658 0.007868 **
## PC12         0.0035853  0.0012632   2.838 0.004538 **
## PC20         0.0043398  0.0014496   2.994 0.002755 **
## PC9          0.0034400  0.0013232   2.600 0.009329 **
## PC5         -0.0018997  0.0007620  -2.493 0.012664 *
## PC10         0.0028300  0.0010981   2.577 0.009963 **
## PC4         -0.0011987  0.0006533  -1.835 0.066548 .
## PC14        -0.0020457  0.0012475  -1.640 0.101038
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2913.25  on 2103  degrees of freedom
## Residual deviance:  142.65  on 2088  degrees of freedom
## AIC: 174.65
##
## Number of Fisher Scoring iterations: 10
```

```
forward_glm$anova
```

```
##      Step Df    Deviance Resid. Df Resid. Dev       AIC
## 1          NA          NA      2103  2913.2471 2915.2471
## 2   + PC1 -1 2616.350401      2102   296.8967  300.8967
## 3   + PC6 -1   45.647348      2101   251.2494  257.2494
## 4   + PC7 -1   27.478649      2100   223.7707  231.7707
## 5   + PC8 -1   14.408248      2099   209.3625  219.3625
## 6  + PC16 -1    9.569141      2098   199.7934  211.7934
## 7  + PC11 -1   10.733620      2097   189.0597  203.0597
## 8  + PC13 -1   12.397102      2096   176.6626  192.6626
## 9  + PC15 -1    9.337983      2095   167.3247  185.3247
## 10 + PC12 -1    5.194648      2094   162.1300  182.1300
## 11 + PC20 -1    3.116652      2093   159.0134  181.0134
## 12  + PC9 -1    3.870922      2092   155.1424  179.1424
## 13  + PC5 -1    3.400437      2091   151.7420  177.7420
## 14 + PC10 -1    3.545717      2090   148.1963  176.1963
## 15  + PC4 -1    2.788664      2089   145.4076  175.4076
## 16 + PC14 -1    2.760514      2088   142.6471  174.6471
```

```
formules_forward <- NULL
formules_forward[[1]] <- "label ~ PC1"
for(i in 3:nrow(forward_glm$anova)){
   formules_forward[[i-1]] <- paste0(formules_forward[[i-2]], forward_glm$ano
va[i,1])
}

formules_forward <- lapply(formules_forward, as.formula)
```

```
tikslumai_forward <- NULL
for(i in 1:length(formules_forward)){
   glm <- glm(formula = formules_forward[[i]], data = data_pcs, family = "bin
omial")
   prob_pred = predict(glm, type = "response", newdata = data_pcs %>% select(
-label))
   y_pred = ifelse(prob_pred > 0.5, 3, 1) %>%
      as.factor()
   confusion_matrix <- confusionMatrix(y_pred, data_pcs$label)
   tikslumai_forward[[i]] <- classAcc(confusion_matrix)
}


tikslumai_forward_data <- do.call(rbind.data.frame, tikslumai_forward) %>%
   bind_cols(kiek_pca = 1:length(formules_forward))
colnames(tikslumai_forward_data) <- c("skaitmuo_1", "skaitmuo_3", "bendras",
"kiek_pca")


tikslumai_forward_data %>%
   tidyr::pivot_longer(c(-kiek_pca), names_to = "Kieno Tikslumas", values_to
= "Tikslumas") %>%
   ggplot(aes(x = kiek_pca, y = Tikslumas, color = `Kieno Tikslumas`)) +
   geom_line() +
   theme_minimal() +
   xlab("Pagrindinių Komponenčių Kiekis") +
   ylab("Tikslumas (%)")

ggsave(filename = "tikslumas_pca_forward.png", width = 14, height = 7, units
= "in", bg = "white")
```



Forward metodas įtraukė tik 15 pagrindinių komponenčių. Forward metodas naudojo AIC reikšmę atrinkti reikšmingiems stebėjimams.

8 užduotis

```
forward_glm$anova$Step
```

```
##  [1] ""       "+ PC1"  "+ PC6"  "+ PC7"  "+ PC8"  "+ PC16" "+ PC11" "+ PC1
3"
##  [9] "+ PC15" "+ PC12" "+ PC20" "+ PC9"  "+ PC5"  "+ PC10" "+ PC4"  "+ PC1
4"
```

Ne, forward metodu įtraukiamos pagrindinės komponentės yra ne eilės tvarka, komponentės trauktos tokia eilės tvarka:

PC1 -> PC6 -> PC7 -> PC8 -> PC16 -> PC13 -> PC15 -> PC12 -> PC20 -> PC9 -> PC5 -> PC10 -> PC4 -> PC14

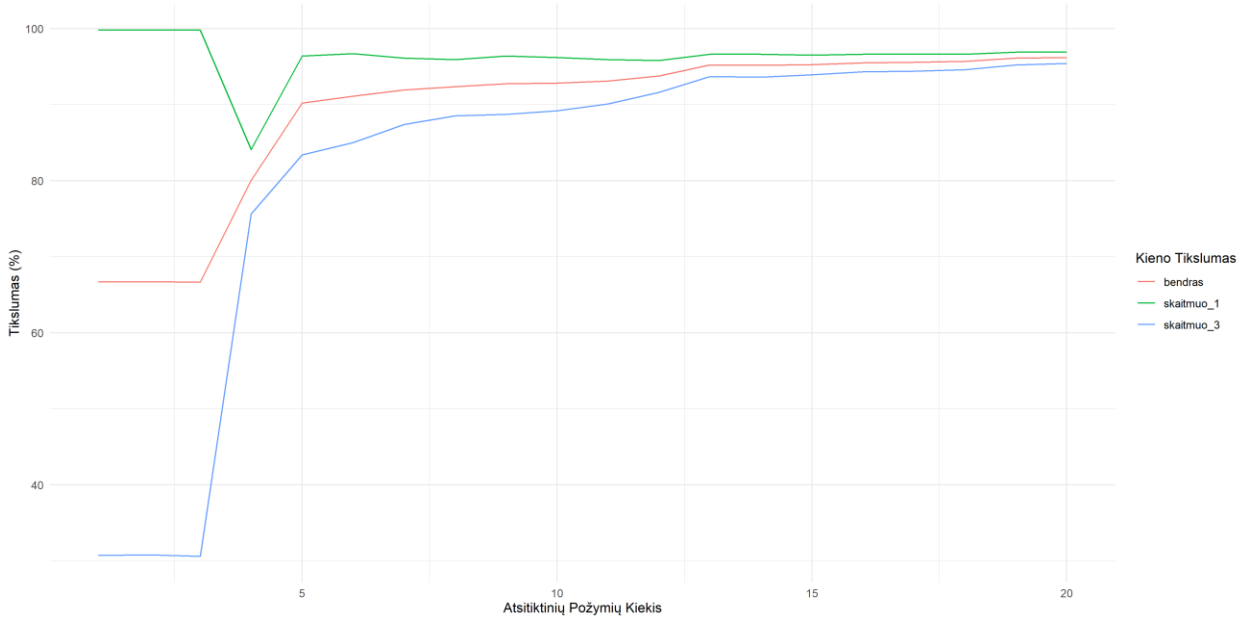
Įdomu, kad metodas neįtraukė nei PC2, nei PC3 komponentės.

9 užduotis

```
atsitiktines_originalios_poros <- sample(names(uzpildyti_pixeliai), 20, repla
ce = F)

formules_originalios <- NULL
formules_originalios[[1]] <- paste0("label ~ ", atsitiktines_originalios_poro
s[1])
for(i in 2:length(atsitiktines_originalios_poros)){
   formules_originalios[[i]] <- paste0(formules_originalios[[i-1]], "+", atsi
tiktines_originalios_poros[i])
}

formules_originalios <- lapply(formules_originalios, as.formula)

tikslumai_originalios <- NULL
for(i in 1:length(formules_originalios)){
   glm <- glm(formula = formules_originalios[[i]], data = data, family = "bin
omial")
   prob_pred = predict(glm, type = "response", newdata = data %>% select(-lab
el))
   y_pred = ifelse(prob_pred > 0.5, 3, 1) %>%
      as.factor()
   confusion_matrix <- confusionMatrix(y_pred, data$label)
   tikslumai_originalios[[i]] <- classAcc(confusion_matrix)
}


tikslumai_originalios_data <- do.call(rbind.data.frame, tikslumai_originalios
) %>%
   bind_cols(kiek_pozymiu = 1:length(formules_originalios))
colnames(tikslumai_originalios_data) <- c("skaitmuo_1", "skaitmuo_3", "bendra
s", "kiek_pozymiu")
```

```
tikslumai_originalios_data %>%
   tidyr::pivot_longer(c(-kiek_pozymiu), names_to = "Kieno Tikslumas", values
_to = "Tikslumas") %>%
   ggplot(aes(x = kiek_pozymiu, y = Tikslumas, color = `Kieno Tikslumas`)) +
   geom_line() +
   theme_minimal() +
   xlab("Atsitiktinių Požymių Kiekis") +
   ylab("Tikslumas (%)")
```

```
ggsave(filename = "tikslumas_originalios.png", width = 14, height = 7, units
= "in", bg = "white")
```



Atsitiktinius pikselius rinkome tik iš pikselių sąrašo, kurie turėjo bent >10% užpildytų (>0) stebėjimų (kitu atveju patektų daug pikselių, kurie neturi jokios informacijos (jų dispersija = 0).

Atrinkus 20 atsitiktinių pikselių ir sudarius 20 logistinės regresijos modelius (pvz 1: label ~ pixel1, 2: label ~ pixel1 + pixel 5) ir t.t., gavome viršutinį grafiką.

10 užduotis

```
glm <- glm(label ~ ., data = data %>% select(all_of(atsitiktines_originalios_
poros), label), family = "binomial")
glm_tuscias <- glm(label ~ 1, data = data, family = "binomial")
# Forward
originalios_forward_glm <- step(glm_tuscias, direction = "forward", scope = f
ormula(glm), trace=0)
summary(originalios_forward_glm)

##
## Call:
## glm(formula = label ~ pixel465 + pixel374 + pixel231 + pixel566 +
##     pixel576 + pixel294 + pixel379 + pixel153 + pixel653 + pixel581 +
##     pixel682 + pixel160 + pixel679 + pixel208 + pixel298, family = "binomi
al",
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.6793  -0.1459  -0.0333   0.0265   2.7637
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.161338   0.393487  -2.951 0.003163 **
## pixel465     0.023228   0.002029  11.447  < 2e-16 ***
```

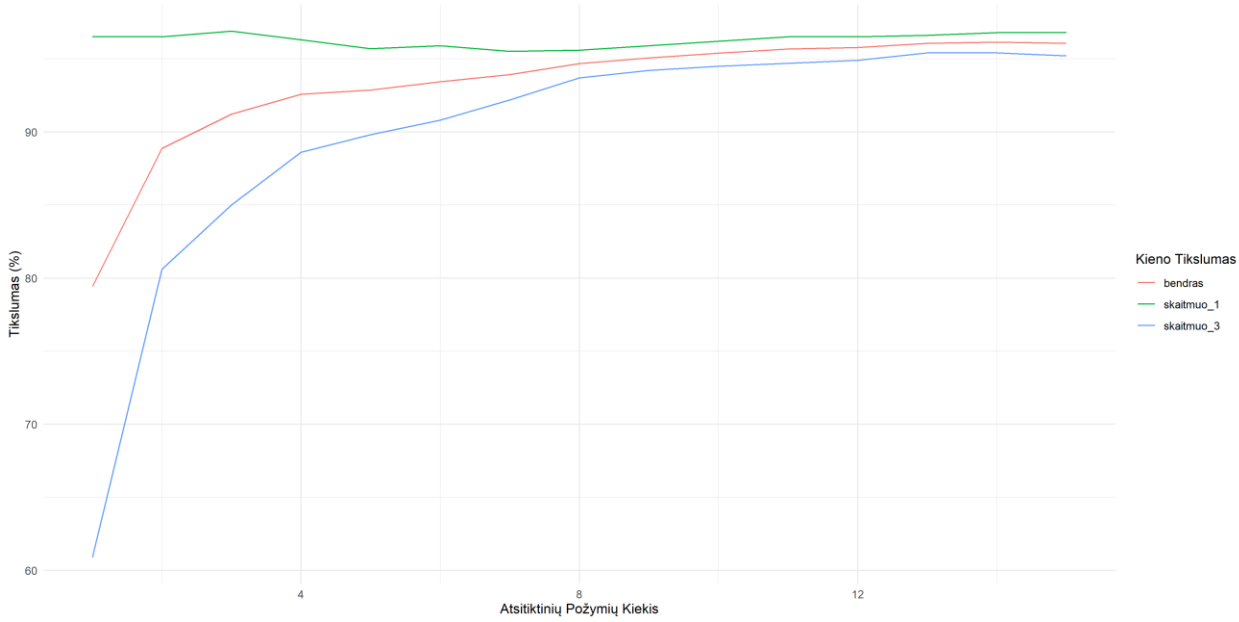
```
## pixel374     0.019398   0.002347   8.265  < 2e-16 ***
## pixel231     0.043406   0.007901   5.494 3.94e-08 ***
## pixel566     0.016655   0.002737   6.086 1.16e-09 ***
## pixel576     0.009425   0.001436   6.563 5.28e-11 ***
## pixel294    -0.011864   0.001532  -7.743 9.70e-15 ***
## pixel379    -0.013235   0.001507  -8.782  < 2e-16 ***
## pixel153     0.009900   0.001412   7.013 2.33e-12 ***
## pixel653     0.007573   0.001677   4.516 6.30e-06 ***
## pixel581     0.022174   0.004296   5.161 2.45e-07 ***
## pixel682     0.007399   0.001792   4.130 3.63e-05 ***
## pixel160    -0.008603   0.002309  -3.726 0.000194 ***
## pixel679     0.007149   0.002888   2.475 0.013328 *
## pixel208    -0.002863   0.001392  -2.057 0.039699 *
## pixel298     0.003095   0.001660   1.864 0.062299 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2913.2  on 2103  degrees of freedom
## Residual deviance:  465.8  on 2088  degrees of freedom
## AIC: 497.8
##
## Number of Fisher Scoring iterations: 9
```

```
originalios_forward_glm$anova
```

```
##          Step Df   Deviance Resid. Df Resid. Dev       AIC
## 1             NA         NA      2103  2913.2471 2915.2471
## 2  + pixel465 -1 951.811642      2102  1961.4355 1965.4355
## 3  + pixel374 -1 665.141353      2101  1296.2942 1302.2942
## 4  + pixel231 -1 215.045339      2100  1081.2488 1089.2488
## 5  + pixel566 -1 154.920136      2099   926.3287  936.3287
## 6  + pixel576 -1  85.484613      2098   840.8441  852.8441
## 7  + pixel294 -1  88.227055      2097   752.6170  766.6170
## 8  + pixel379 -1  89.211270      2096   663.4057  679.4057
## 9  + pixel153 -1  60.410358      2095   602.9954  620.9954
## 10 + pixel653 -1  46.547777      2094   556.4476  576.4476
## 11 + pixel581 -1  42.865065      2093   513.5825  535.5825
## 12 + pixel682 -1  21.356897      2092   492.2256  516.2256
## 13 + pixel160 -1  11.887309      2091   480.3383  506.3383
## 14 + pixel679 -1   6.478036      2090   473.8603  501.8603
## 15 + pixel208 -1   4.547572      2089   469.3127  499.3127
## 16 + pixel298 -1   3.513351      2088   465.7994  497.7994
```

```
formules_originalios_forward <- NULL
formules_originalios_forward[[1]] <- paste0("label ~ ", originalios_forward_g
lm$anova[2,1])
for(i in 3:nrow(originalios_forward_glm$anova)){
   formules_originalios_forward[[i-1]] <- paste0(formules_originalios_forward
[[i-2]], originalios_forward_glm$anova[i,1])
}

formules_originalios_forward <- lapply(formules_originalios_forward, as.formu
la)
```

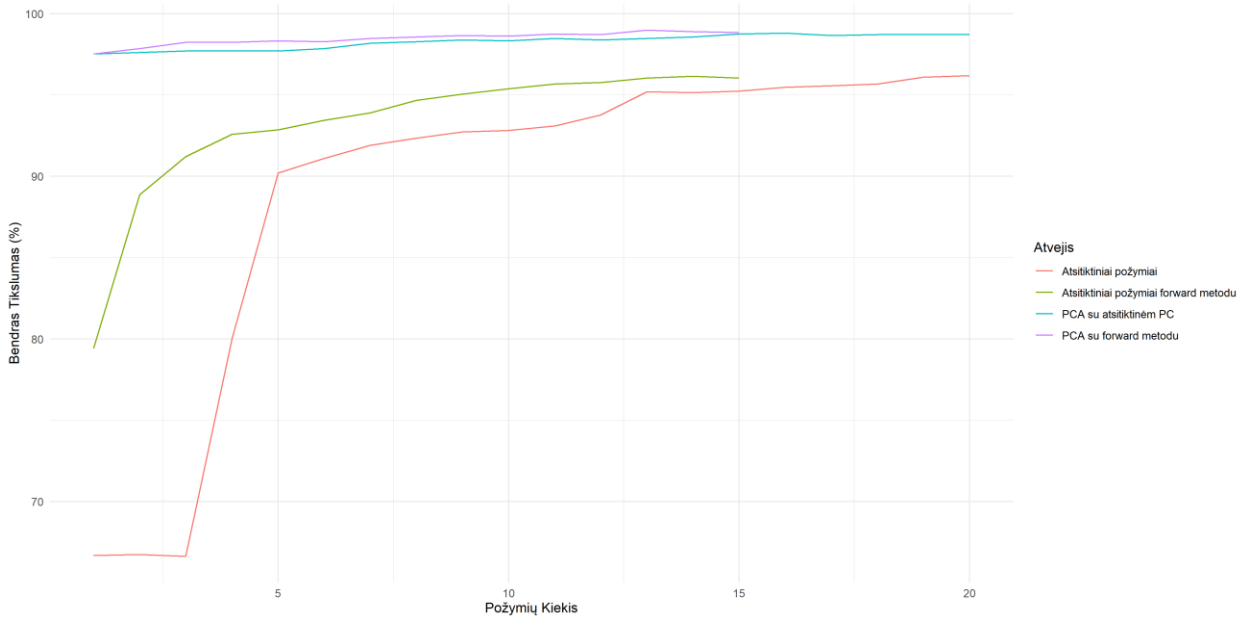
```
tikslumai_originalios_forward <- NULL
for(i in 1:length(formules_originalios_forward)){
   glm <- glm(formula = formules_originalios_forward[[i]], data = data, family = "binomial")
   prob_pred = predict(glm, type = "response", newdata = data %>% select(-label))
   y_pred = ifelse(prob_pred > 0.5, 3, 1) %>%
      as.factor()
   confusion_matrix <- confusionMatrix(y_pred, data$label)
   tikslumai_originalios_forward[[i]] <- classAcc(confusion_matrix)
}


tikslumai_originalios_forward_data <- do.call(rbind.data.frame, tikslumai_originalios_forward) %>%
   bind_cols(kiek_pozymiu = 1:length(formules_originalios_forward))
colnames(tikslumai_originalios_forward_data) <- c("skaitmuo_1", "skaitmuo_3", "bendras", "kiek_pozymiu")


tikslumai_originalios_forward_data %>%
   tidyr::pivot_longer(c(-kiek_pozymiu), names_to = "Kieno Tikslumas", values_to = "Tikslumas") %>%
   ggplot(aes(x = kiek_pozymiu, y = Tikslumas, color = `Kieno Tikslumas`)) +
   geom_line() +
   theme_minimal() +
   xlab("Atsitiktinių Požymių Kiekis") +
   ylab("Tikslumas (%)")

ggsave(filename = "tikslumas_originalios_forward.png", width = 14, height = 7, units = "in", bg = "white")
```



Forward metodas atrinko 15 reikšmingų atsitiktinių požymių.

11 užduotis

```
tikslumai_visi <- data.frame(
   Bendras_Tikslumas = c(tikslumai_data$bendras, tikslumai_forward_data$bendr
as, tikslumai_originalios_data$bendras, tikslumai_originalios_forward_data$be
ndras),
   Kiek_Pozymiu = c(tikslumai_data$kiek_pca, tikslumai_forward_data$kiek_pca,
tikslumai_originalios_data$kiek_pozymiu, tikslumai_originalios_forward_data$k
iek_pozymiu),
   Atvejis = c(rep("PCA su atsitiktinėm PC", nrow(tikslumai_data)), rep("PCA
su forward metodu", nrow(tikslumai_forward_data)), rep("Atsitiktiniai požymia
i", nrow(tikslumai_originalios_data)), rep("Atsitiktiniai požymiai forward me
todu", nrow(tikslumai_originalios_forward_data)))
)

tikslumai_visi %>%
   ggplot(aes(x = Kiek_Pozymiu, y = Bendras_Tikslumas, color = Atvejis)) +
   geom_line() +
   theme_minimal() +
   xlab("Požymių Kiekis") +
   ylab("Bendras Tikslumas (%)")

ggsave(filename = "tikslumas_visu.png", width = 14, height = 7, units = "in",
bg = "white")
```

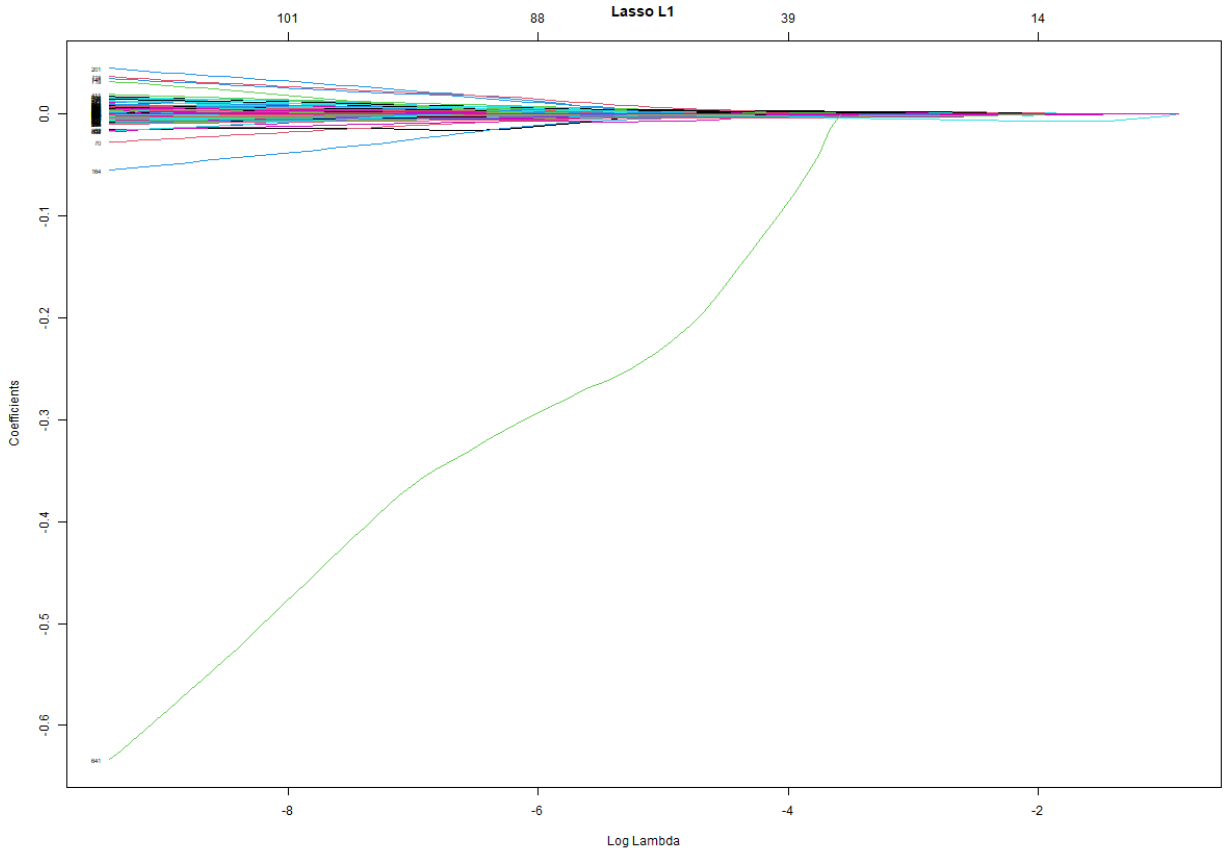


Akivaizdžiai matosi, jog naudojant PCA visais atvejais gaunami aukštesni tikslumai, nei naudojant atsitiktinius pikselius. Abejais atvejais, forward metodas atrinko 15 reikšmingų požymių.

12 užduotis

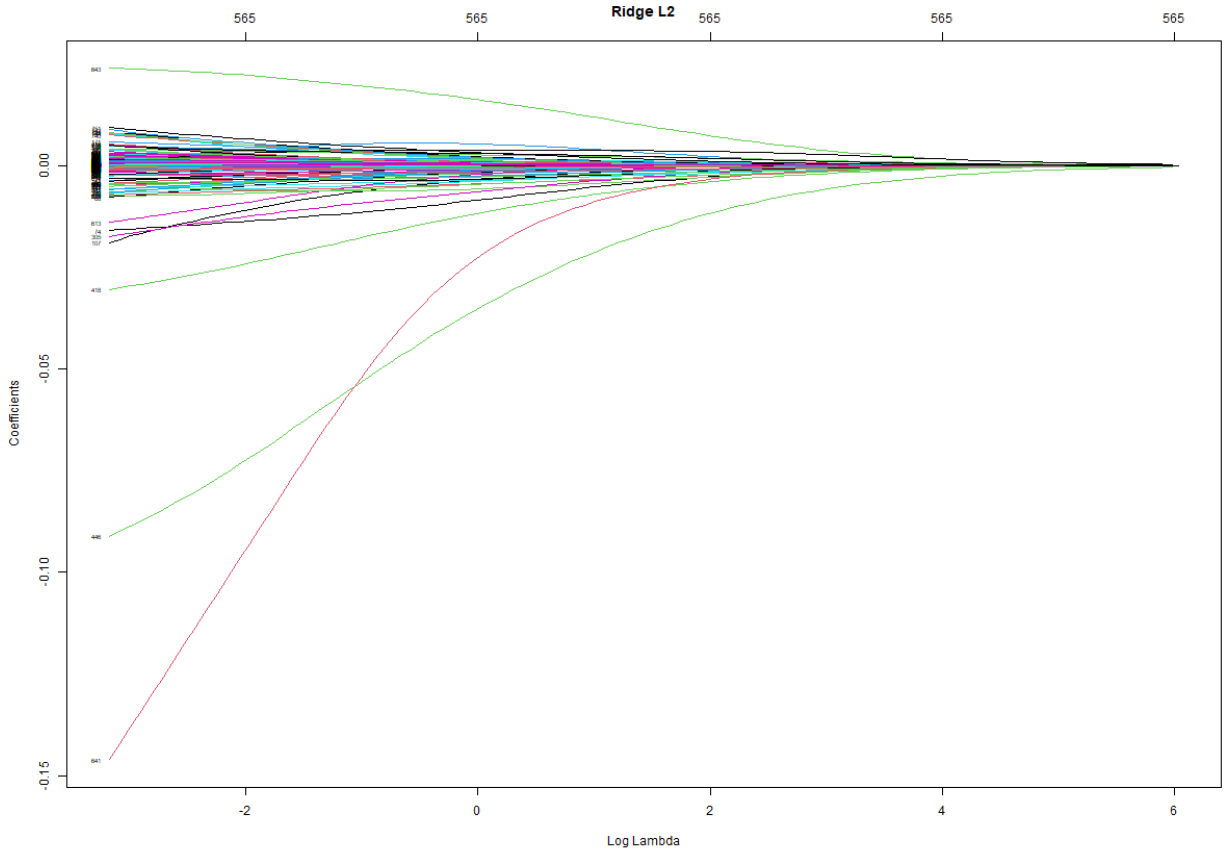
```
lasso_L1 <- glmnet(data %>% select(-label), data$label, family = "binomial",
alpha = 1)
ridge_L2 <- glmnet(data %>% select(-label), data$label, family = "binomial",
```

```
alpha = 0)

png(file = "lasso_l1.png", width = 1200, height = 850)
plot(lasso_L1, label=TRUE,xvar="lambda", main="Lasso L1")
dev.off()
```



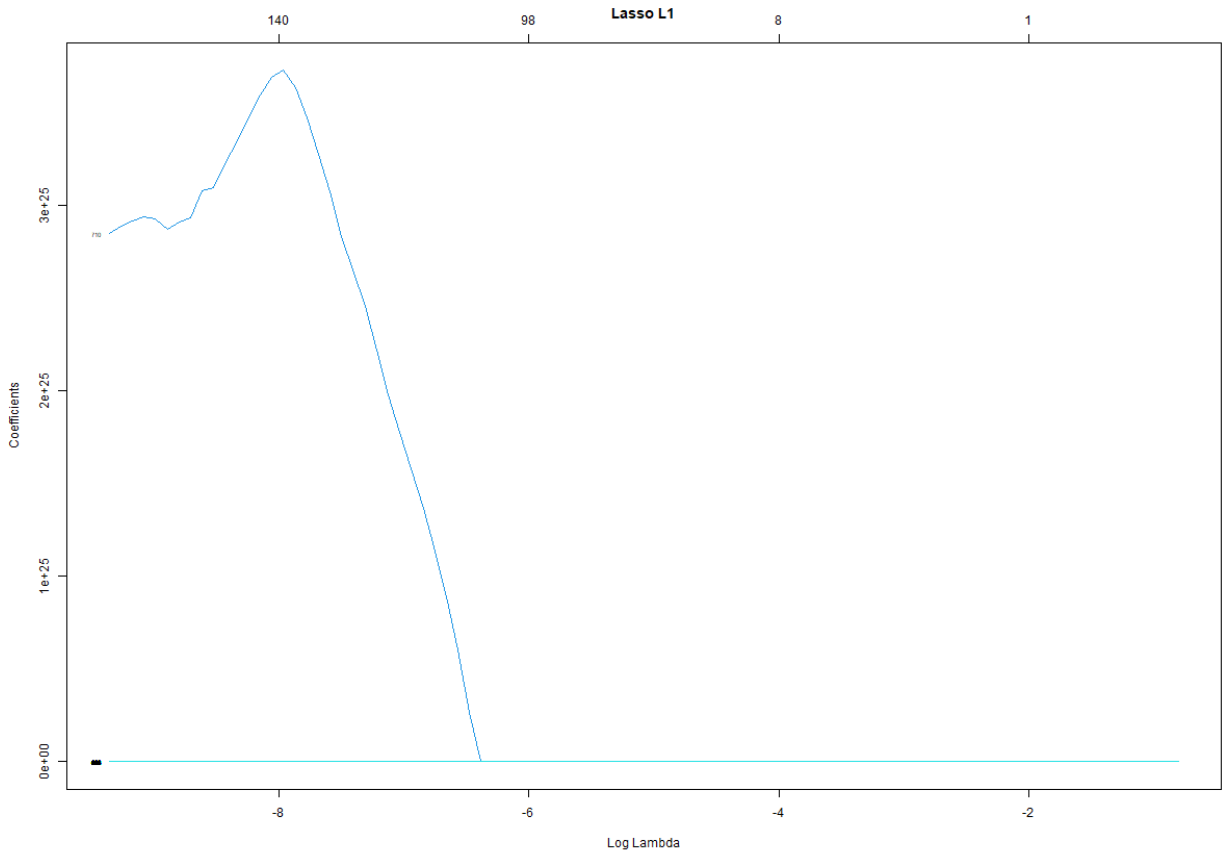
Koeficientų priklausomybė nuo lasso L1 reguliarizavimo konstantos lambda kitimo grafikai kiekvienai iš reguliarizacijų kuomet naudojam visus pradinius požymius.

```
png(file = "ridge_l2.png", width = 1200, height = 850)
plot(ridge_L2,label=TRUE,xvar="lambda", main="Ridge L2")
dev.off()
```



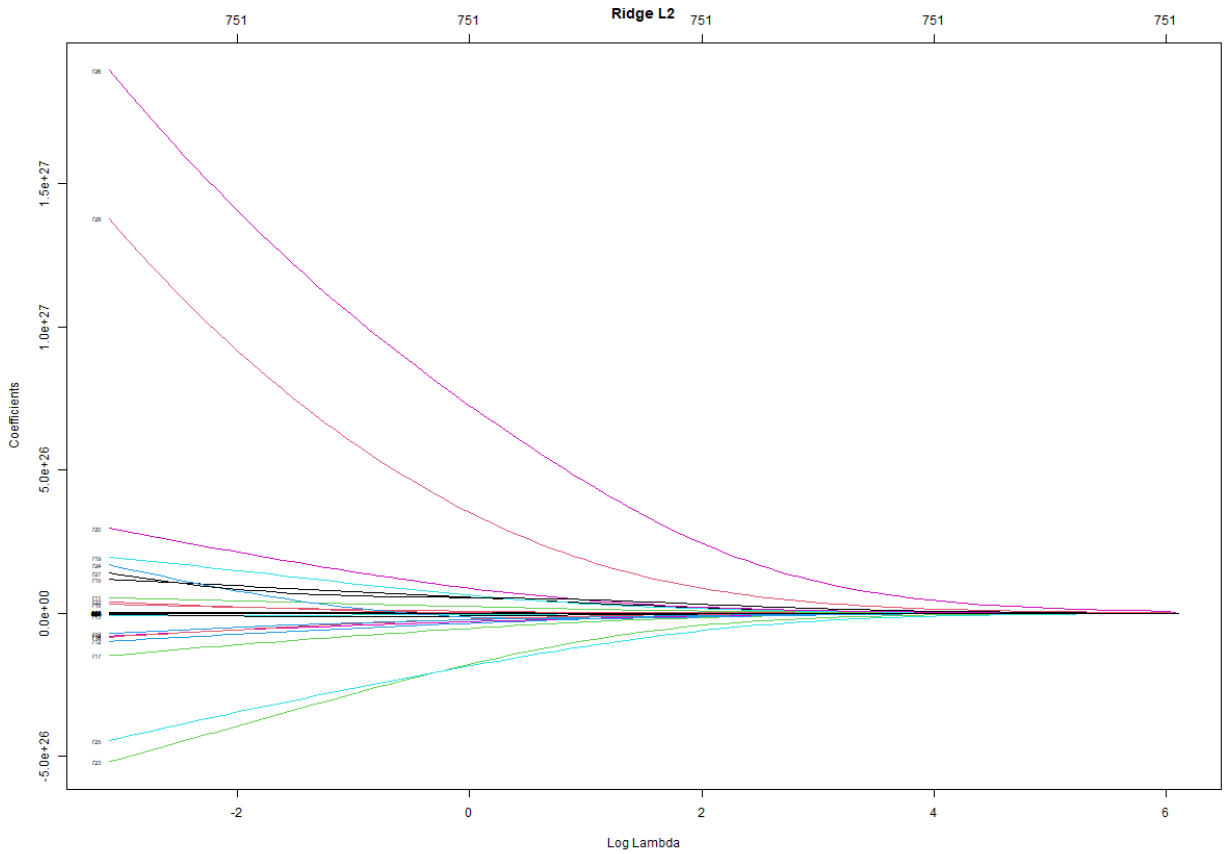
Koeficientų priklausomybė nuo ridge L1 reguliarizavimo konstantos lambda kitimo grafikai kiekvienai iš reguliarizacijų kuomet naudojam visus pradinius požymius.

```
lasso_L1_pca <- glmnet(data_pcs %>% select(-label), data_pcs$label, family =
"binomial", alpha = 1)
ridge_L2_pca <- glmnet(data_pcs %>% select(-label), data_pcs$label, family =
"binomial", alpha = 0)

png(file = "lasso_l1_pca.png", width = 1200, height = 850)
plot(lasso_L1_pca, label=TRUE,xvar="lambda", main="Lasso L1")
dev.off()
```



Koeficientų priklausomybė nuo lasso L1 reguliarizavimo konstantos lambda kitimo grafikai kiekvienai iš reguliarizacijų kuomet naudojam visas pagrindines komponentes.

```
png(file = "ridge_l2_pca.png", width = 1200, height = 850)
plot(ridge_L2_pca, label=TRUE,xvar="lambda", main="Ridge L2")
dev.off()
```



Koeficientų priklausomybė nuo ridge L1 reguliarizavimo konstantos lambda kitimo grafikai kiekvienai iš reguliarizacijų kuomet naudojam visas pagrindines komponentes.

13 užduotis

```
lambdas <- lasso_L1$lambda

lambda <- NULL

for(i in 1:length(lambdas)){
   lassoCoef = coef(lasso_L1, s = lambdas[i])
   nonzero_coef = lassoCoef[lassoCoef[,1] != 0,]
   if(length(nonzero_coef) - 1 == 10){
      lambda <- lambdas[i]
      break
   }
}



lassoCoef <- coef(lasso_L1, s = lambda)
lassoCoef <- lassoCoef[lassoCoef[,1] != 0,]
lambda

## [1] 0.1498659

lassoCoef
```

```
##   (Intercept)      pixel179      pixel375      pixel409      pixel462
##  6.670124e-01  8.914077e-04  8.535919e-04  7.285605e-04 -9.622494e-04
##      pixel489      pixel490      pixel494      pixel517      pixel522
## -7.135796e-03 -7.732842e-04  4.888314e-04 -6.193543e-05  5.149765e-04
##      pixel550
##  4.514332e-04
```

```
lambdas <- lasso_L1_pca$lambda

lambda_pca <- NULL

for(i in 1:length(lambdas)){
   lassoCoef = coef(lasso_L1_pca, s = lambdas[i])
   nonzero_coef = lassoCoef[lassoCoef[,1] != 0,]
   if(length(nonzero_coef) - 1 == 10){
      lambda_pca <- lambdas[i]
      break
   }
}



lassoCoef <- coef(lasso_L1_pca, s = lambda_pca)
lassoCoef <- lassoCoef[lassoCoef[,1] != 0,]
lambda_pca

## [1] 0.01575323

lassoCoef

##   (Intercept)           PC1           PC6           PC7           PC8
##  4.543868e-01 -5.777272e-03  1.076897e-03 -9.677499e-05 -3.201329e-04
##          PC11          PC13          PC30         PC565         PC591
##  4.236738e-04 -1.635797e-04 -1.236879e-04 -7.053515e+11  3.061386e+11
##         PC623
##  1.065933e+11
```

Originalių požymiu atveju:

Konstanta - 0.1498659

Ir su ja įtraukti 10 požymiai – pixel179, pixel375, pixel409, pixel462, pixel489, pixel490, pixel494, pixel517, pixel522, pixel550.

Pagrindinių komponenčių atveju:

Konstanta - 0.01575323

Ir su ja įtrauktos 10 komponenčių – PC1, PC6, PC7, PC8, PC11, PC13, PC30, PC565, PC591, PC623.

LITERATŪRA

1. Buvo naudotos tik paketų dokumentacijos.