Príloha A: Obsah elektronického média

K práci je priložený kompaktný disk, na ktorom sa nachádzajú nasledovné súbory:

- Súbor *bakalarska_praca.pdf*, ktorý je elektronickou formou tohto dokumentu.
- Adresár *SQL*, ktorý obsahuje textové súbory s SQL príkazmi SQL/PL funkciami použitými na správu databázy a import dát do databázy.
- Adresár *R_source*, ktorý obsahuje skripty v jazyku R používané na predikciu, experimenty a testovanie predikcie.
- Adresár *ostatne*, ktorý obsahuje textové súbory s SQL príkazmi a skripty v jazyku R, ktoré nie sú súčasťou finálnej verzie, ale slúžili ako pomocné SQL príkazy alebo pomocné skripty pri práci.
- Súbor *popis.txt*, ktorý obsahuje popis adresárov a súborov, ktoré sa nachádzajú na elektronickom médiu.

Poznámka: Dáta, ktoré sme pre predikciu použili nie sú obsahom elektronického média, pretože dáta so záznamami o produkcii fotovoltaických elektrární nemáme dovolené ďalej sprostredkovať alebo zverejňovať. Tým pádom je zbytočné prikladať dáta so záznamami predpovede počasia, pretože bez dát o produkcii FVE nie je možné použiť zdrojové kódy našej implementácie.

Príloha B: Technická dokumentácia

Naše riešenie predikcie produkcie fotovoltaickej elektrárne sme implementovali v jazyku R vo verzii 3.2.3 vydanej 10.12.2015. Požili sme niekoľko balíkov z verejného repozitára CRAN, ktoré nie sú súčasťou verzie 3.2.3. Použité balíky: randomForest, RPostgreSQL, insol, sirad, plyr, snow, compiler, microbenchmark.

Neimplementovali sme stabilnú aplikáciu alebo systém, ale skripty vykonávajúce predikciu a overenie presnosti predikcie. Dáta, s ktorými pracujeme pri predikcii, sme uložili do relačnej databázy PostgreSQL vo verzii 9.5.

Nasledujúci zdrojový kód v jazyku R je poslednou verziou skriptu, ktorým vykonávame predikciu a overujeme jej presnosť. V postupnosť krokov v zdrojovom kóde:

1. načítanie používaných balíkov a funkcii z externého súboru,
2. pripravenie paralelizácie na štyroch jadrách a exportovanie funkcie *format.time* a balíkov *randomForest* a *plyr* na všetky jadrá,
3. inicializácia premenných používaných na nastavenie parametrov predikčného modelu a výberu dát do trénovacej množiny,
4. vytvorenie karteziánskeho súčinu všetkých hodnôt nastavení,
5. výpis času začatia vykonávania skriptu,
6. vytvorenie spojenia s databázou,
7. inicializácia a výpočet hodnôt premenných používaných na výpis stavu predikcie,
8. cyklus pre každý riadok z tabuľky karteziánskeho súčinu nastavení,
9. cyklus pre každú z elektrární,
10. vytiahnutie záznamov z databázy, výpočet škály pre hodnotenie podobnosti záznamov,
11. exportovanie premenných na všetky jadrá procesora, paralelný výpočet predikcie pre každý záznam zvlášť, vráti vektor predikovaných hodnôt pre jednu elektráreň,
12. vymedzenie potenciálnych záznamov do trénovacej množiny,
13. ohodnotenie potenciálnych záznamov podľa podobnosti,
14. výber najpodobnejších záznamov do trénovacej množiny,
15. vytvorenie náhodného lesa regresných stromov,
16. predikcia produkcie pre vybraný záznam,
17. pridanie skutočných a predikovaných hodnôt do vektorov spoločných pre všetky elektrárne,
18. výpis stavu predikcie,
19. výpočet hodinovej štatistiky presnosti a zápis výsledkov do databázy,

20. úprava výsledkov pre výpočet dennej štatistiky presnosti,
21. výpočet dennej štatistiky presnosti a zápis výsledkov do databázy,
22. konečný výpis,
23. ukončenie možnosti paralelizácie a zatvorenie spojenia s databázou.

Zdrojový kód 1: Skript pre predikciu a výpočet štatistiky presnosti.

```
# 1. načítanie používaných balíkov a funkcii z externého súboru
library(RPostgreSQL)
library(plyr)
library(randomForest)
library(snow)
source('~/GitHub/baka/R source/new_way/functions.R')
# 2. pripravenie paralelizácie na štyroch jadrách a exportovanie funkcie
# format.time a balí-kov randomForest a plyr na všetky jadrá
cl <- makeCluster(4, type='SOCK')
clusterEvalQ(cl, format.time <- function(x) UseMethod("format.time"))
clusterEvalQ(cl, { library(plyr); library(randomForest) })
# 3. inicializácia premenných používaných na nastavenie parametrov
# predikčného modelu a výberu dát do trénovacej množiny
write_results <- TRUE
dataset <- c("v_data_all", "v_data_120", "v_data", "v_data_vz",
"v_data_vz_120")
fve <- c(1, 2, 3)
tm_velkost <- c(30)
f_ntree <- c(500)
f_mtry <- c(2)
f_nodesize <- c(3)
pod_gho <- c(210)
pod_obl <- c(90)
pod_tep <- c(7.5)
pod_vie <- c(5.5)
pod_vlh <- c(3.5)
pod_dlz <- c(59)
pod_ele <- c(41)
ele_res <- c(6)
select <- " SELECT datum, cas, praca, gho, oblacnost,
            teplota, vietor, vlhkost, dlzkadna, elev
            FROM %s WHERE fve = %d ORDER BY cas"
# 4. vytvorenie karteziánskeho súčinu všetkých hodnôt nastavení
settings <- expand.grid(dataset = dataset, velkost = tm_velkost,
    ntree = f_ntree, mtry = f_mtry, nodesize = f_nodesize,
    pod_gho = pod_gho, pod_obl = pod_obl, pod_tep = pod_tep,
    pod_vie = pod_vie, pod_vlh = pod_vlh, pod_dlz = pod_dlz,
    pod_ele = pod_ele, ele_res = ele_res)
# 5. výpis času začatia vykonávania skriptu
time.start <- Sys.time()
print(sprintf("Start: %s ", time.start), quote = F)
# 6. vytvorenie spojenia s databázou
db.drv <- dbDriver("PostgreSQL")
if (exists("db.con")) dbDisconnect(db.con)
db.con <- getConnection(db.drv)
# 7. inicializácia a výpočet hodnôt premenných používaných na výpis
prog.diff <- 0
prog.printed_all <- -10000
prog.printed_ops <- -10000
prog.print_perc_all <- 0
prog.baseAll <- 0
```

```
prog.opsAll <- 0
prog.i <- 0
prog.op <- 0
prog.opsAll <- nrow(settings)
prog.baseAll <- 0
for (i.dataset in dataset) {
  b_select <- "select count(*) as ccc from (select distinct * from
                    (select cas, fve from %s where fve IN (%s)) s1) s2"
  prog.baseAll <- prog.baseAll + dbGetQuery(db.con,
           sprintf(b_select, i.dataset, toString(fve)))$ccc
}
prog.baseAll <- prog.baseAll * prog.opsAll / length(dataset)
hours_done <- 0
ops_done <- 0
# 8. cyklus pre každý riadok z tabuľky karteziánskeho súčinu nastavení
for (i.sett in 1:prog.opsAll) {
  setting <- settings[i.sett,]
  ops_done <- i.sett
  actual <- c()
  output <- c()
  # 9. cyklus pre každú z elektrární
  for (i.fve in fve) {
    # 10.   vytiahnutie záznamov z databázy,
    # výpočet škály pre hodnotenie podobnosti záznamov
    all_hours <- dbGetQuery(db.con, sprintf(select,
                                          setting$dataset, i.fve))
    ad_ncol <- ncol(all_hours)
    maxims <- apply(all_hours[,4:ad_ncol], 2, max)
    minims <- apply(all_hours[,4:ad_ncol], 2, min)
    scale <- abs(maxims - minims)
    all_hours <- data.matrix(all_hours)
    chosen_hours <- all_hours
    # 11. exportovanie premenných na všetky jadrá procesora, paralelný
      výpočet predikcie
    # pre každý záznam zvlášť, vráti vektor predikovaných hodnôt pre jednu
      elektráreň
    clusterExport(cl, list("chosen_hours", "all_hours", "scale",
                                                          "setting"))
    fve_output <- parSapply(cl, 1:nrow(chosen_hours), function(y) {
      # 12. vymedzenie potenciálnych záznamov do trénovacej množiny
      hourh <- chosen_hours[y,]
      potencial <- all_hours[all_hours[,'datum'] != hourh[['datum']],]
      potencial <- potencial[abs(potencial[,'elev'] - hourh[['elev']]) <=
                                                        setting$ele_res,]
      # 13. ohodnotenie potenciálnych záznamov podľa podobnosti
      diff <- vector(mode = "numeric", length = nrow(potencial))
      diff <- sapply(1:length(diff), function(x) {
        ret <- abs(hourh[['gho']] - potencial[[x,'gho']])
                              * 100 / scale[['gho']] * setting$pod_gho
        ret <- ret + abs(hourh[['oblacnost']] - potencial[[x,'oblacnost']])
                              * 100 / scale[['oblacnost']] * setting$pod_obl
        ret <- ret + abs(hourh[['teplota']] - potencial[[x,'teplota']])
                              * 100 / scale[['teplota']] * setting$pod_tep
        ret <- ret + abs(hourh[['vietor']] - potencial[[x,'vietor']])
                              * 100 / scale[['vietor']] * setting$pod_vie
        ret <- ret + abs(hourh[['vlhkost']] - potencial[[x,'vlhkost']])
                              * 100 / scale[['vlhkost']] * setting$pod_vlh
        ret <- ret + abs(hourh[['dlzkadna']] - potencial[[x,'dlzkadna']])
                              * 100 / scale[['dlzkadna']] * setting$pod_dlz
        ret <- ret + abs(hourh[['elev']] - potencial[[x,'elev']])
                              * 100 / scale[['elev']] * setting$pod_ele
```

```
            return(ret)})
      # 14. výber najpodobnejších záznamov do trénovacej množiny
      train_set <- arrange(as.data.frame(potencial), diff)
      [1:setting$velkost,]
      # 15. vytvorenie náhodného lesa regresných stromov
      forest <- randomForest(
            praca~gho+oblacnost+teplota+vietor+vlhkost+dlzkadna+elev,
            data=train_set, ntree = setting$ntree, mtry = setting$mtry,
            nodesize = setting$nodesize)
      # 16. predikcia produkcie pre vybraný záznam
      predic <-predict(forest, data.frame(gho = hourh[['gho']],
                                          oblacnost = hourh[['oblacnost']],
                                          teplota = hourh[['teplota']],
                                          vietor = hourh[['vietor']],
                                          vlhkost = hourh[['vlhkost']],
                                          dlzkadna = hourh[['dlzkadna']],
                                          elev = hourh[['elev']]),
                       type="response", norm.votes=TRUE)
      return(predic)
    }) # koniec paralelizovaného kódu
    # 17. pridanie skutočných a predikovaných hodnôt do vektorov spoločných
      # pre všetky elektrárne
    fve_actual <- chosen_hours[,'praca']
    actual <- append(actual, fve_actual)
    output <- append(output, fve_output)
    # 18. výpis stavu predikcie
    hours_done <- hours_done + nrow(all_hours)
    prog.i <- hours_done
    prog.print_perc_all <- (prog.i * 100 / prog.baseAll)
    prog.op <- ops_done
    prog.print_perc_ops <- (prog.op * 100 / prog.opsAll)
    if (prog.print_perc_all > prog.printed_all + prog.diff) {
      prog.actual_time <- as.numeric(difftime(Sys.time(),
                                              time.start, units = "sec"))
      prog.estimated_time <- prog.actual_time * 100 / prog.print_perc_all
      print(sprintf("Forest perc: %6.2f%s, ops: %7.d/%d, day: %9.d/%d,
                    Estimated time: %s, Actual: %s", #
                    prog.print_perc_all, "%", prog.op, prog.opsAll, prog.i,
                    prog.baseAll, format.time(prog.estimated_time),
                    format.time(prog.actual_time)),
            quote=F)
      prog.printed_all <- prog.print_perc_all
    }
  } # koniec cyklu pre každú elektráreň
  # 19.  výpočet štatistiky presnosti
  stats <- all_statistics(actual, output)
  if (write_results) {
    for (name in names(stats)) {
      if (is.infinite(stats[[name]]) | !is.numeric(stats[[name]]) |
                                                  is.nan(stats[[name]]))
        stats[[name]] <- 999.999
    }
    insert <- sprintf("INSERT INTO t_experiment (cas_behu, metoda, param1,
                      param2, param3, param4, param5, N, MBE, RMBE, RMSE,
                      RRMSE, MAE, RMAE, MPE, MAXAE, SD, tm_velkost,
                      tm_opis, tm_select, fve, den_hod, pod_gho,
                      pod_oblacnost, pod_teplota, pod_vietor,
                      pod_vlhkost, pod_tlak, pod_dlzkadna, pod_azim,
                      pod_elev, in_gho, in_oblacnost, in_teplota,
                      in_vietor, in_vlhkost, in_tlak, in_dlzkadna,
```

```
                        in_azim, in_elev) VALUES ('%s', '%s', '%s', '%s',
                        '%s', '%s', '%s', %d, %f, %f, %f, %f, %f, %f, %f,
                        %f, %f, %d, '%s', '%s', '%s', '%s', %f, %f, %f, %f,
                    %f, %f, %f, %f, %f, %s, %s, %s, %s, %s, %s, %s, %s, %s);",
                        time.start, "stats_hod", setting$dataset, "ntree "
                        %s% as.character(setting$ntree) , "mtry " %s%
                        as.character(setting$mtry), "nodesize " %s%
                        as.character(setting$nodesize), " ", stats$N,
                        stats$MBE, stats$RMBE, stats$RMSE, stats$RRMSE,
                        stats$MAE, stats$RMAE, stats$MPE, stats$MAXAE,
                        stats$SD, setting$velkost, "30 najpodob hodin",
                        select, toString(fve), "hod", setting$pod_gho,
                        setting$pod_obl, setting$pod_tep, setting$pod_vie,
                        setting$pod_vlh, 0, setting$pod_dlz, 0,
                        setting$pod_ele, TRUE, TRUE, TRUE, TRUE, TRUE,
                        FALSE, TRUE, FALSE, TRUE)
    db.result <- dbGetQuery(db.con, insert)
  }
# 20.   úprava výsledkov pre výpočet dennej štatistiky presnosti
  s_select <- "SELECT fve, datum, cas, gho, oblacnost,
                      teplota, vietor, vlhkost, dlzkadna, elev, praca
                      FROM %s WHERE fve IN (%s) ORDER BY fve, cas"
  all_data <-  dbGetQuery(db.con, sprintf(s_select, setting$dataset,
                                                    toString(fve)))
  all_data <- cbind(all_data, output)
  to_see <- cbind(all_data, dif = abs(output - actual) * 100 / actual)
  to_see <- arrange(to_see, to_see$dif)
  groupped <- ddply(all_data,~datum+fve,summarise, gho=sum(gho),
                      oblacnost=sum(oblacnost), teplota=sum(teplota),
                      vietor=sum(vietor), vlhkost=sum(vlhkost),
                      dlzkadna=max(dlzkadna), praca=sum(praca),
                      output=sum(output))
  # 21. výpočet dennej štatistiky presnosti a zápis výsledkov do databázy
  stats_day <- all_statistics(groupped$praca, groupped$output)
  if (write_results) {
    for (name in names(stats_day)) {
      if (is.infinite(stats_day[[name]]) | !is.numeric(stats_day[[name]]) |
                                              is.nan(stats_day[[name]]))
          stats_day[[name]] <- 999.999
    }
    insert <- sprintf("INSERT INTO t_experiment (cas_behu, metoda, param1,
                                  param2, param3, param4, param5,
                                   N, MBE, RMBE, RMSE, RRMSE, MAE, RMAE,
                                  MPE, MAXAE, SD, tm_velkost, tm_opis,
                                  tm_select, fve, den_hod, pod_gho,
                                  pod_oblacnost, pod_teplota, pod_vietor,
                                  pod_vlhkost, pod_tlak, pod_dlzkadna,
                                  pod_azim, pod_elev, in_gho,
                                  in_oblacnost, in_teplota, in_vietor,
                                  in_vlhkost, in_tlak, in_dlzkadna,
                                  in_azim, in_elev) VALUES ('%s', '%s',
                                  '%s', '%s', '%s', '%s', '%s', %d, %f,
                                  %f, %f, %f, %f, %f, %f, %f, %f, %d,
                                  '%s', '%s', '%s', '%s', %f, %f, %f, %f,
                                  %f, %f, %f, %f, %f, %s, %s, %s, %s, %s,
                                  %s, %s, %s, %s);", time.start,
                                  "stats_den", setting$dataset, "ntree "
                                  %s% as.character(setting$ntree) , "mtry
                                  " %s%, as.character(setting$mtry),
```

```
                        "nodesize " %s% as.character(setting$nodesize),
                        " ", stats_day$N, stats_day$MBE, stats_day$RMBE,
                        stats_day$RMSE, stats_day$RRMSE, stats_day$MAE,
                        stats_day$RMAE, stats_day$MPE, stats_day$MAXAE,
                        stats_day$SD, setting$velkost, "30 najpodob hodin",
                        select, toString(fve), "hod", setting$pod_gho,
                        setting$pod_obl, setting$pod_tep, setting$pod_vie,
                        setting$pod_vlh, 0, setting$pod_dlz, 0,
                        setting$pod_ele, TRUE, TRUE, TRUE, TRUE, TRUE,
                        FALSE, TRUE, TRUE, FALSE)
    db.result <- dbGetQuery(db.con, insert)
  }
} # koniec cyklu pre každý riadok z tabuľky karteziánskeho súčinu nastavení
# 22. konečný výpis,
time.end <- Sys.time()
print(sprintf("Start: %s, End: %s, Duration: %s",
              time.start, time.end,
              format.time(difftime(time.end, time.start, units = "sec"))),
      quote = F)
# 23. ukončenie možnosti paralelizácie a zatvorenie spojenia s databázou.
stopCluster(cl)
if (exists("db.con")) dbDisconnect(db.con)
```

V nasledujúcom zdrojovom kóde implementácia vlastných funkcií použitých v predchádzajúcom skripte.

Zdrojový kód 2: Implementácia vlastných funkcií v externom súbore.

```
# vráti spojenie s databázou
getConnection <- function(drv) {
  con <- dbConnect(drv, dbname = "test_db", host = "localhost",
                   port = 5432, user = "postgres", password = "password")
  return(con)
}
# výpočet štatistiky presnosti
library(Metrics)
library(sirad)
all_statistics <- function(actual, predicted) {
  mdval <- modeval(predicted, actual,stat = c("N", "MBE", "RMBE", "RMSE",
                  "RRMSE", "MAE", "RMAE", "MPE", "SD"), minlength = 2)
  mdval$MAXAE <- max(ae(actual, predicted))
  return(mdval)
}
# formátovanie času
format.time <- function(x) UseMethod("format.time")
format.time.difftime <- function(x) {
  units(x) <- "secs"
  x <- unclass(x)
  NextMethod()
}
format.time.default <- function(x) {
  y <- abs(x)
  sprintf("%s%02dh:%02dm:%02ds", ifelse(x < 0, "-", ""),
      y %% 86400 %/% 3600, y %% 3600 %/% 60, y %% 60 %/% 1)
}
# vlastný operátor na spájanie reťazcov
`%s%` <- function(s1, s2) paste0(s1, s2)
```