R Notebook

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

```
library(ggplot2)
library(xgboost)
library(caret)
```

```
## Loading required package: lattice
```

```
library(dplyr)
```

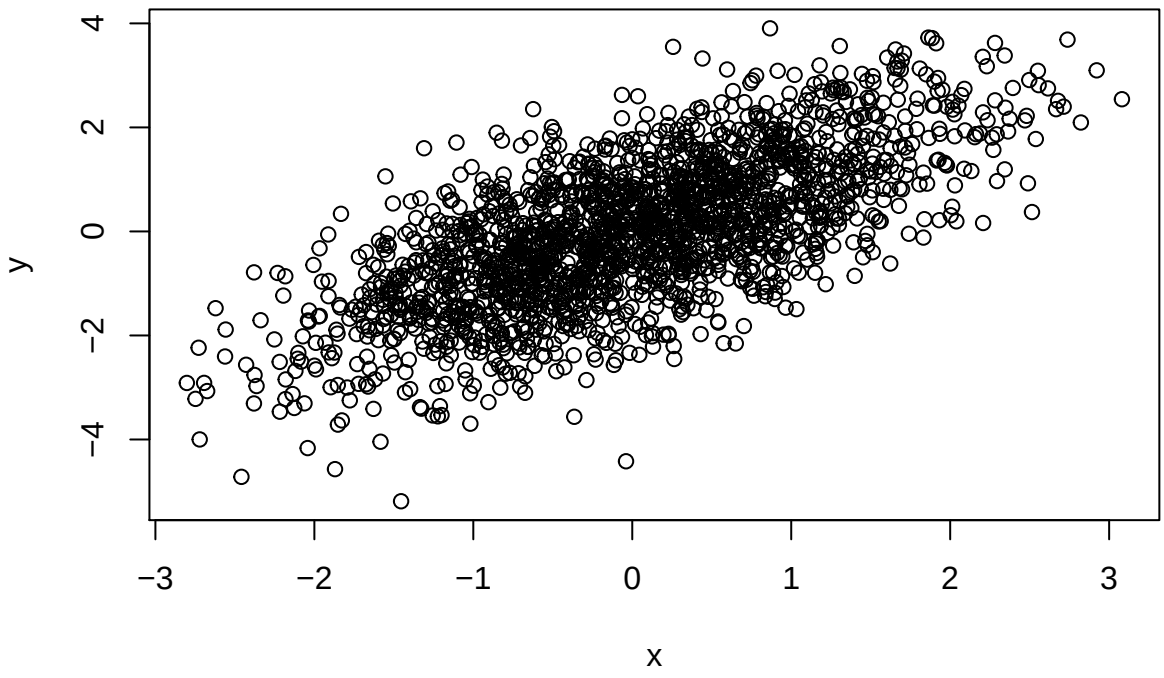
```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:xgboost':
##
##     slice
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
x <- rnorm(2000)
y <- x + rnorm(2000)
plot(x,y)
```




```
df <- data.frame(x,y)

train <- df %>% filter(x<0.5)
test <- df %>% filter(x>=0.5)
```

```
train_x <- as.matrix(select(train, x))
train_y <- select(train, y)[[1]]

test_x <- as.matrix(select(test, x))
test_y <- select(test, y)[[1]]


dtrain <- xgb.DMatrix(data = train_x,label=train_y)
dtest <- xgb.DMatrix(data = test_x)
```

```
# Params for xgboost
param <- list(objective="reg:linear",
              booster="gbtree",
              eval_metric = "rmse",
              eta = .05,
              gamma = 1,
              max_depth = 1,
              min_child_weight = 1,
              subsample = 1,
              colsample_bytree = 1
)
```

```
#Cross validation - determine CV scores & optimal amount of rounds
xgb_cv <- xgb.cv(data = dtrain,
                 nfold = 5,
                 params = param,
                 nrounds = 150000,
                 maximize = FALSE,
                 prediction = TRUE,
                 early_stopping_round = 100,
                 verbose=0
)

rounds <- xgb_cv$best_iteration

# Train model
cat("XGB training")
```

```
## XGB training
```

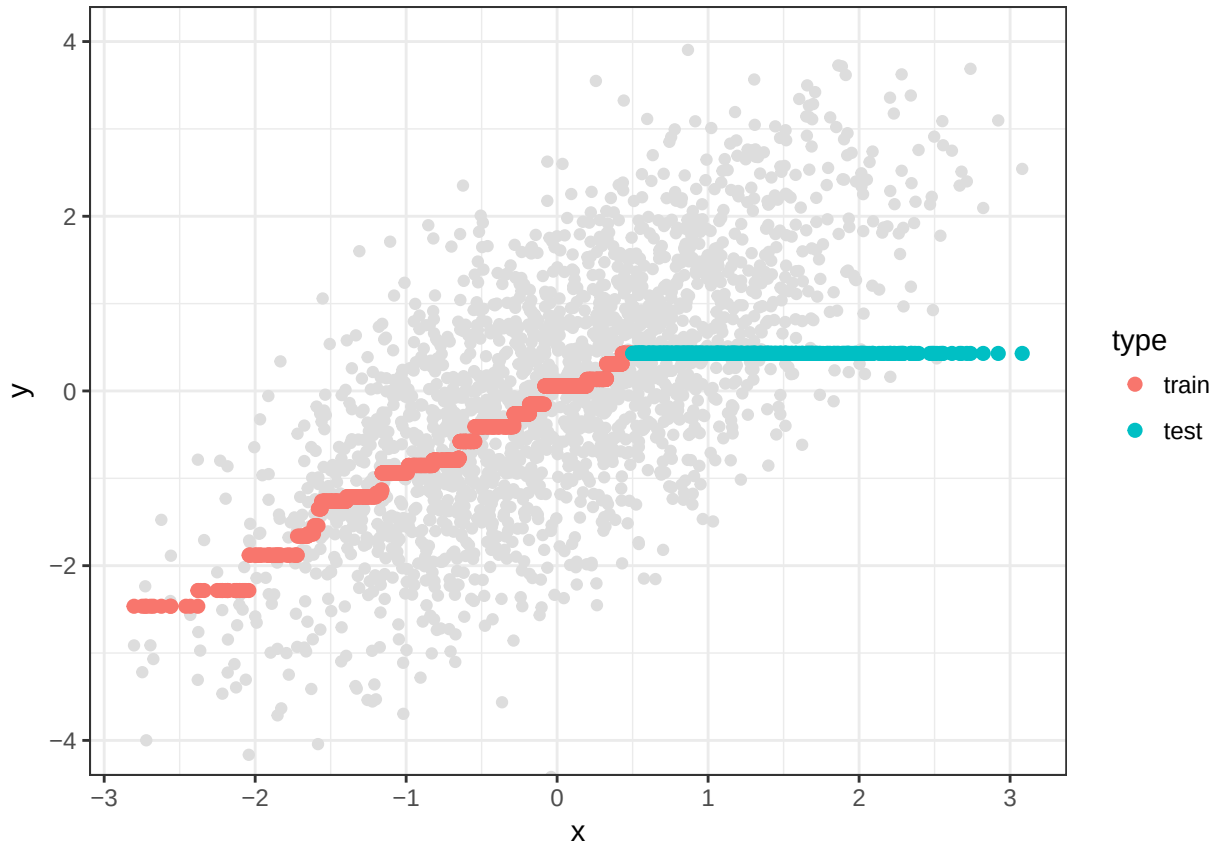
```
xgb_model <- xgb.train(data = dtrain,
                       params = param,
                       watchlist = list(train = dtrain),
                       nrounds = rounds,
                       verbose = 0
)

preds_train <- predict(xgb_model,dtrain)
preds_test <- predict(xgb_model,dtest)

plot_train <- data.frame(x=train_x,y=train_y,pred=preds_train,type=0)
plot_test <- data.frame(x=test_x,y=test_y,pred=preds_test,type=1)

plotdf <- bind_rows(plot_train,plot_test)
plotdf$type = factor(plotdf$type,labels = c("train","test"))
```

```
plotdf %>% ggplot(aes(x=x,color=type))+geom_point(aes(y=y), color="#DDDDDD")+theme_bw()+coord_cartesian
```




```
cat('\nTrain RMSE: ', RMSE(plot_train$y,plot_train$pred))
```

```
##
## Train RMSE:  1.002085
```

```
cat('\nCV RMSE: ', xgb_cv$evaluation_log$test_rmse_mean[rounds])
```

```
##
## CV RMSE:  1.027764
```

```
cat('\nTest RMSE: ', RMSE(plot_test$y,plot_test$pred))
```

```
##
## Test RMSE:  1.288673
```

```
# Params for xgboost
param <- list(objective="reg:linear",
              booster="gblinear",
              eval_metric = "rmse",
              eta = .05,
              gamma = 1,
              max_depth = 1,
              min_child_weight = 1,
              subsample = 1,
              colsample_bytree = 1
)
```

```r
#Cross validation - determine CV scores & optimal amount of rounds
xgb_cv <- xgb.cv(data = dtrain,
                 nfold = 5,
                 params = param,
                 nrounds = 150000,
                 maximize = FALSE,
                 prediction = TRUE,
                 early_stopping_round = 100,
                 verbose=0
)

rounds <- xgb_cv$best_iteration

# Train model
cat("XGB training")
```

```
## XGB training
```

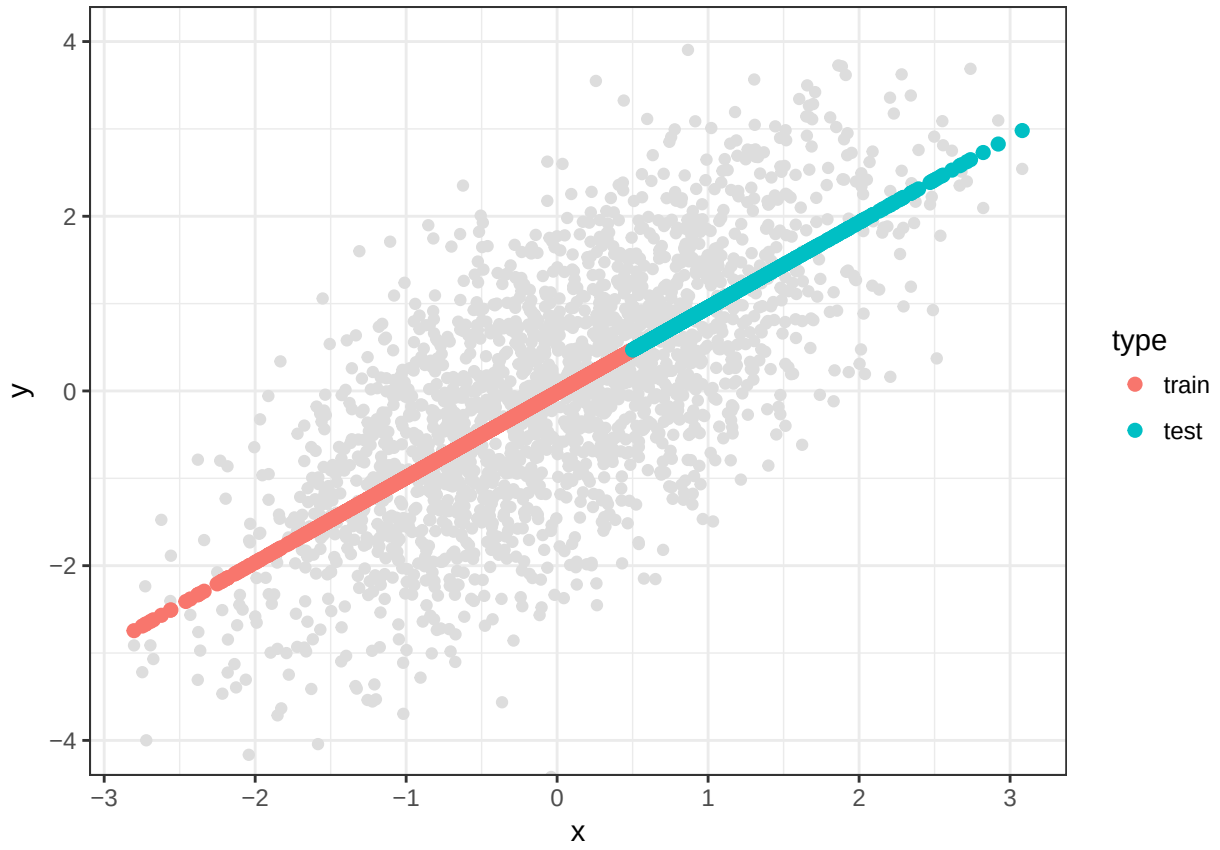
```r
xgb_model <- xgb.train(data = dtrain,
                       params = param,
                       watchlist = list(train = dtrain),
                       nrounds = rounds,
                       verbose = 0
)

preds_train <- predict(xgb_model,dtrain)
preds_test <- predict(xgb_model,dtest)

plot_train <- data.frame(x=train_x,y=train_y,pred=preds_train,type=0)
plot_test <- data.frame(x=test_x,y=test_y,pred=preds_test,type=1)

plotdf <- bind_rows(plot_train,plot_test)
plotdf$type = factor(plotdf$type,labels = c("train","test"))
```

```r
plotdf %>% ggplot(aes(x=x,color=type))+geom_point(aes(y=y), color="#DDDDDD")+theme_bw()+coord_cartesian
```




```
cat('\nTrain RMSE: ', RMSE(plot_train$y,plot_train$pred))
```

```
##
## Train RMSE:  1.013673
```

```
cat('\nCV RMSE: ', xgb_cv$evaluation_log$test_rmse_mean[rounds])
```

```
##
## CV RMSE:  1.013799
```

```
cat('\nTest RMSE: ', RMSE(plot_test$y,plot_test$pred))
```

```
##
## Test RMSE:  1.000001
```

```
inds <- sample(1:nrow(train),nrow(train)*0.8)
train_y[inds] <- 2

dtrain <- xgb.DMatrix(data = train_x,label=train_y)
```

```
# Params for xgboost
param <- list(objective="reg:linear",
              booster="gblinear",
              eval_metric = "rmse",
              eta = .05,
              gamma = 1,
              max_depth = 1,
              min_child_weight = 1,
              subsample = 1,
```

```
              colsample_bytree = 1
)


#Cross validation - determine CV scores & optimal amount of rounds
xgb_cv <- xgb.cv(data = dtrain,
                 nfold = 5,
                 params = param,
                 nrounds = 150000,
                 maximize = FALSE,
                 prediction = TRUE,
                 early_stopping_round = 100,
                 verbose=0
)

rounds <- xgb_cv$best_iteration

# Train model
cat("XGB training")
```

```
## XGB training
```

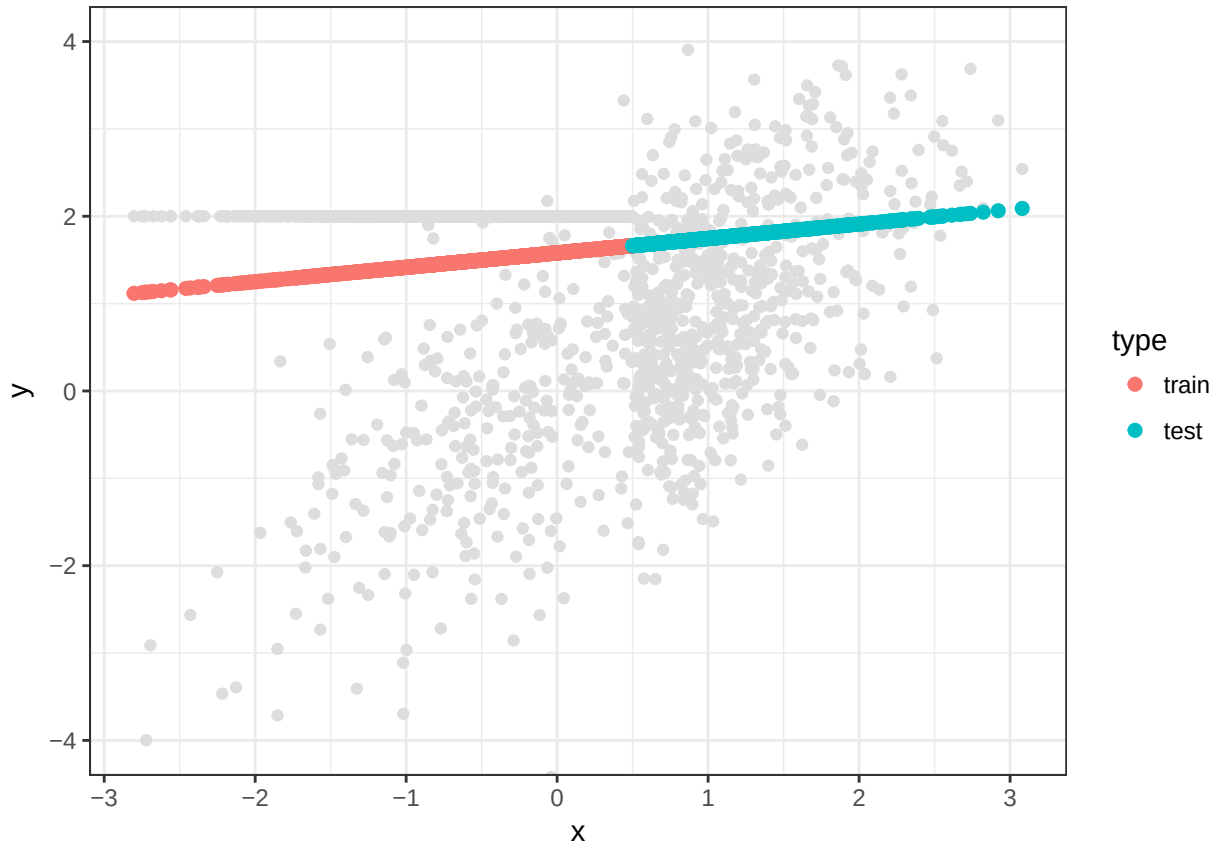
```
xgb_model <- xgb.train(data = dtrain,
                       params = param,
                       watchlist = list(train = dtrain),
                       nrounds = rounds,
                       verbose = 0
)

preds_train <- predict(xgb_model,dtrain)
preds_test <- predict(xgb_model,dtest)

plot_train <- data.frame(x=train_x,y=train_y,pred=preds_train,type=0)
plot_test <- data.frame(x=test_x,y=test_y,pred=preds_test,type=1)

plotdf <- bind_rows(plot_train,plot_test)
plotdf$type = factor(plotdf$type,labels = c("train","test"))
```

```
plotdf %>% ggplot(aes(x=x,color=type))+geom_point(aes(y=y), color="#DDDDDD")+theme_bw()+coord_cartesian
```




```
cat('\nTrain RMSE: ', RMSE(plot_train$y,plot_train$pred))
```

```
##
## Train RMSE:  1.138425
```

```
cat('\nCV RMSE: ', xgb_cv$evaluation_log$test_rmse_mean[rounds])
```

```
##
## CV RMSE:  1.136595
```

```
cat('\nTest RMSE: ', RMSE(plot_test$y,plot_test$pred))
```

```
##
## Test RMSE:  1.291571
```