# Grid Search

#how do i now which parameter we should select during machine learning model

#how do i know which model choose for my machine learning problem.

#here aim is to improve model performance

#in ML,first type of parameter is parameter it learn during machine learning and second type of parameter is parameter we choose i.e hyperparameter ex kernal,svm model, penalty parameter and some regularization parameter.

#

#-----------------------------

#1)look at dependent variable, is its have continous outcome or categorical outcome///--

#it continous outcome then problem is regression problem

#if it is categorical outcome then it have classification problem.

#dont have dependent variable then it is clustering problem

#2) my problem is linear problem or non-linear problem,, for large amount of dataset its defficult to figureout data is linearly seperable or rather choose linear model like SVM if doing classification or non-linear model if doing kernal-SVM,, this question cab be answered by technique called grid search

# Importing the dataset

dataset = read.csv('Social\_Network\_Ads.csv')

dataset = dataset[3:5]

# Encoding the target feature as factor

dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))

# Splitting the dataset into the Training set and Test set

# install.packages('caTools')

library(caTools)

set.seed(123)

split = sample.split(dataset$Purchased, SplitRatio = 0.75)

training\_set = subset(dataset, split == TRUE)

test\_set = subset(dataset, split == FALSE)

# Feature Scaling

training\_set[-3] = scale(training\_set[-3])

test\_set[-3] = scale(test\_set[-3])

# Fitting Kernel SVM to the Training set

# install.packages('e1071')

library(e1071)

classifier = svm(formula = Purchased ~ .,

data = training\_set,

type = 'C-classification',

kernel = 'radial',

sigma = 2.251496, # add here from bestTune value in grid search of caret package(below)

C =1) # add here from bestTune value in grid search of caret package(below)

# Predicting the Test set results

y\_pred = predict(classifier, newdata = test\_set[-3])

# Making the Confusion Matrix

cm = table(test\_set[, 3], y\_pred)

cm

# Applying k-Fold Cross Validation

# install.packages('caret')

library(caret)

folds = createFolds(training\_set$Purchased, k = 10)

cv = lapply(folds, function(x) {

training\_fold = training\_set[-x, ]

test\_fold = training\_set[x, ]

classifier = svm(formula = Purchased ~ .,

data = training\_fold,

type = 'C-classification',

kernel = 'radial')

y\_pred = predict(classifier, newdata = test\_fold[-3])

cm = table(test\_fold[, 3], y\_pred)

accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])

return(accuracy)

})

accuracy = mean(as.numeric(cv))

#---------------------------

#for parameter tuning we should use caret package

#http://topepo.github.io/caret/available-models.html

#-----------------------------

# Applying Grid Search to find the best parameters

# install.packages('caret')

library(caret) # our model is "Support Vector Machines with Radial Basis Function Kernel

classifier = train(form = Purchased ~ ., data = training\_set, method = 'svmRadial')# press F1

classifier #C, accuracy, kappa

classifier$bestTune #to get optimal value of hyperparameter

#========================================

#you can use caret build classifier or take bestTune Hypperparameter from caret package classifier and build your own classifier (SVM classifier)above

#==========================================

# Visualising the Training set results

library(ElemStatLearn)

set = training\_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set)

plot(set[, -3],

main = 'Kernel SVM (Training set)',

xlab = 'Age', ylab = 'Estimated Salary',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

# Visualising the Test set results

library(ElemStatLearn)

set = test\_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set)

plot(set[, -3], main = 'Kernel SVM (Test set)',

xlab = 'Age', ylab = 'Estimated Salary',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))