World Bank

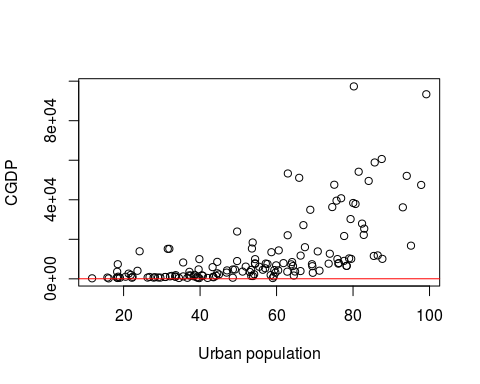
### Introduction

The objective of this project was analyzing the data of the World Bank to predict the urban population in the data set using three different regression models and analyzing which one would predict the urban population accurately.

### Linear Model

My first analysis was plotting urban population as a function of cgdp. I then constucted a linear model but the results were barely acceptable.

# Plot urb\_pop as function of cgdp  
plot(world\_bank\_train$urb\_pop , world\_bank\_train$cgdp , xlab = "Urban population" , ylab = "CGDP")  
  
# Set up a linear model between the two variables  
lm\_wb = lm(urb\_pop ~ cgdp , data= world\_bank\_train)  
  
# Add a red regression line to your scatter plot  
abline(lm\_wb$coefficient , col = "red")



# Summarize lm\_wb and select R-squared  
summary(lm\_wb)$r.squared

## [1] 0.3822067

In the previous model, the scatter plot didn't show a strong linear relationship. This was confirmed with the regression line and R^2

To improve the linear model we need to study the nature of the data. The predictor variable is numerical, while the response variable is expressed in percentiles. It would make more sense if there were a linear relationship between the percentile changes of the GDP / capita and the changes in the response.

To obtain an estimation of percentile changes, we need to take the natural logarithm of the GDP / capita and use this as your new predictor variable. world\_bank\_test set is used to check if the model generalizes well.

### Log-Linear Model

# Build the log-linear model  
lm\_wb\_log <- lm(urb\_pop ~ log(cgdp), data = world\_bank\_train)  
  
# Calculate rmse\_train  
rmse\_train <- sqrt(mean(lm\_wb\_log$residuals ^ 2))  
  
# Summarize lm\_wb and select R-squared  
summary(lm\_wb\_log)$r.squared

## [1] 0.5787588

# The real percentage of urban population in the test set  
world\_bank\_test\_truth <- world\_bank\_test$urb\_pop  
  
# The predictions of the percentage of urban population in the test set  
world\_bank\_test\_input <- data.frame(cgdp = world\_bank\_test$cgdp)  
world\_bank\_test\_output <- predict(lm\_wb\_log, world\_bank\_test\_input)  
  
# The residuals: the difference between the ground truth and the predictions  
res\_test <- world\_bank\_test\_output - world\_bank\_test\_truth  
  
  
# Use res\_test to calculate rmse\_test  
rmse\_test = sqrt(sum((res\_test)^2)/nrow(world\_bank\_test))  
  
# Print the ratio of the test RMSE over the training RMSE  
rmse\_test/rmse\_train

## [1] 1.08308

### Linear model v/s Log-Linear model

Prediction differed substantially between these two models. R^2 value for Log-Linear model is greater than the R^2 value of Linear model. Therefore Log-Linear model gives better prediction as its explained variance is the highest.

### k-NN regression model

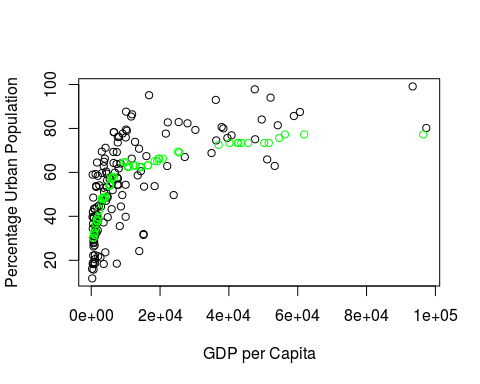
The function my\_knn contains a k-NN algorithm.

It's arguments are:

x\_pred : predictor values of the new observations   
 x : predictor values of the training set   
 y: corresponding response values of the training set   
 k: the number of neighbors

The function returns the predicted values for your new observations (predict\_knn).

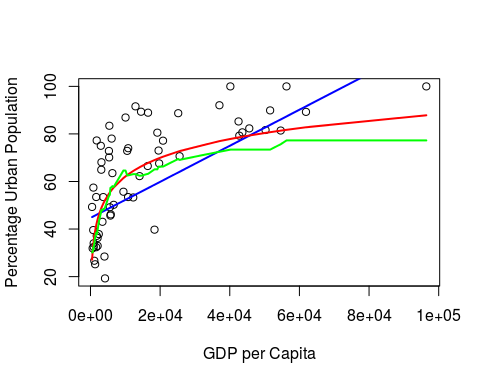
my\_knn <- function(x\_pred, x, y, k){  
 m <- length(x\_pred)  
 predict\_knn <- rep(0, m)  
 for (i in 1:m) {  
   
 # Calculate the absolute distance between x\_pred[i] and x  
 dist <- abs(x\_pred[i] - x)  
   
 # Apply order() to dist, sort\_index will contain   
 # the indices of elements in the dist vector, in   
 # ascending order. This means sort\_index[1:k] will  
 # return the indices of the k-nearest neighbors.  
 sort\_index <- order(dist)   
   
 # Apply mean() to the responses of the k-nearest neighbors  
 predict\_knn[i] <- mean(y[sort\_index[1:k]])   
   
 }  
 return(predict\_knn)  
}  
###  
  
# Applied alogrithm on the test set  
test\_output = my\_knn(world\_bank\_test$cgdp ,world\_bank\_train$cgdp , world\_bank\_train$urb\_pop , 30)  
  
# Plot of the output  
plot(world\_bank\_train,   
 xlab = "GDP per Capita",   
 ylab = "Percentage Urban Population")  
points(world\_bank\_test$cgdp, test\_output, col = "green")



### Linear model v/s Log-Linear model v/s k-NN Regression model

Compare the RMSE of the three models to see which will give the best predictions.

# Define ranks to order the predictor variables in the test set  
ranks <- order(world\_bank\_test$cgdp)  
  
# Scatter plot of test set  
plot(world\_bank\_test,   
 xlab = "GDP per Capita", ylab = "Percentage Urban Population")  
  
# Predict with simple linear model and add line  
test\_output\_lm <- predict(lm\_wb, data.frame(cgdp = world\_bank\_test$cgdp))  
lines(world\_bank\_test$cgdp[ranks], test\_output\_lm[ranks], lwd = 2, col = "blue")  
  
# Predict with log-linear model and add line  
test\_output\_lm\_log <- predict(lm\_wb\_log, data.frame(cgdp = world\_bank\_test$cgdp))  
lines(world\_bank\_test$cgdp[ranks], test\_output\_lm\_log[ranks], lwd = 2, col = "red")  
  
# Predict with k-NN and add line  
test\_output\_knn <- my\_knn(world\_bank\_test$cgdp, world\_bank\_train$cgdp, world\_bank\_train$urb\_pop, 30)  
lines(world\_bank\_test$cgdp[ranks], test\_output\_knn[ranks], lwd = 2, col = "green")



# Calculate RMSE for simple linear model  
sqrt(mean( (test\_output\_lm - world\_bank\_test$urb\_pop) ^ 2))

## [1] 17.41897

# Calculate RMSE for log-linear model  
sqrt(mean( (test\_output\_lm\_log - world\_bank\_test$urb\_pop) ^ 2))

## [1] 15.01911

# Calculate RMSE for k-NN technique  
sqrt(mean( (test\_output\_knn - world\_bank\_test$urb\_pop) ^ 2))

## [1] 16.10026

In conclusion the Log-Linear model gives the best RMSE value out of all the three models and thereby giving the most accurate prediction.