# intro-to-network-helper-functions.r

# These functions make it easier to perform certain EDA with the igraph package, and/or to extract data from the Pitts-Spillane social network data.

```r
node.centrality <- function(G) {
  rbind("In degree" = degree(G,mode="in"),
        "Out degree" = degree(G,mode="out"),
        "Closeness" = round(closeness(G,mode="all"),2),
        "Betweenness" = round(betweenness(G),2))
}

edge.centrality <- function(G) {
  H <- G
  attr(H,"class") <- "list"
  H.names <- H[[9]][[3]]$name
  H.from <- H[[4]] + 1
  H.to <- H[[4]] + 1
  E.list <- cbind(H.names[H.from],H.names[H.to])
  E.list <- apply(E.list,1,paste,collapse=" - ")
  data.frame(bt=round(edge.betweenness(G,2),row.names=E.list))
}
```

```r
table = xtable(node.centrality(G))
print(table, type = "html")
```

```r
table = xtable(edge.centrality(G))
print(table, type = "html")
```

# intro-to-network-helper-functions.r

```r
node.centrality <- function(G) {
  rbind("In degree" = degree(G,mode="in"),
        "Out degree" = degree(G,mode="out"),
        "Closeness" = round(closeness(G,mode="all"),2),
        "Betweenness" = round(betweenness(G),2))
}

edge.centrality <- function(G) {
  H <- G
  attr(H,"class") <- "list"
  H.names <- H[[9]][[3]]$name
  H.from <- H[[4]] + 1
  H.to <- H[[4]] + 1
  E.list <- cbind(H.names[H.from],H.names[H.to])
  E.list <- apply(E.list,1,paste,collapse=" - ")
  data.frame(bt=round(edge.betweenness(G,2),row.names=E.list))
}
```

```r
table = xtable(node.centrality(G))
print(table, type = "html")
```

```r
table = xtable(edge.centrality(G))
print(table, type = "html")
```
intro-to-networks-helper-functions.r

dimnames(teacher.Xs)[[3]],
dimnames(school.Xs)[[3]])
return(X)
}

extract.Y <- function(school) {
  Y <- advice.mat[[school]]
  diag(Y) <- NA
  Y <- c(Y)
  return(Y)
}

########################################################################
# Helper functions to extract Y and X for #
# dyadic independence regression for several #
# networks at once, from the Pitts & Spillane #
# e.g. to extract the data you need to do a #
# joint regression analysis for networks 1, 3, #
# 5 and 9, just type #
# Y <- stack.Y(c(1,3,5,9)) #
# X <- stack.X(c(1,3,5,9)) #
# and then do a logistic regression like this: #
# test.glm <- glm(Y ~ X, family=binomial) #
# summary(test.glm) #
# or if you wanted to build X and Y for all 15 #
# networks, type #
# Y <- stack.Y(1:15) #
# X <- stack.X(1:15) #

stack.X <- function(schools) {
  X <- NULL
  for (i in schools) {
    X <- rbind(X, extract.X(i))
  }
  return(X)
}

stack.Y <- function(schools) {
  Y <- NULL
  for (i in schools) {
    Y <- c(Y, extract.Y(i))
  }
  return(Y)
}

Page 3

Page 4