

Below tables summarize the results we elicited from k-NN and linear regression. As can be seen from the tables, the k-NN performance in training as well as in testing is higher than the linear regression.

	K=1	K=3	K=5	K=7	K=15
test_error_knn	0.02472527	0.02289377	0.02285714	0.02354788	0.02681319
train_error_knn	0.000000000	0.003519718	0.004607631	0.005377529	0.008338533

test_error_bls	0.05397913
train_error_bls	0.02481285

The code we used for the above results are as follows:

```
train=read.csv(file="train.csv",head=FALSE,sep=",")
test=read.csv(file="test.csv",head=FALSE,sep=",")
```

```
bls<-function(x0,x,y){
  x0 <- as.matrix(x0)
  x0%*%solve(t(x)%*%x,t(x)%*%y)
}
```

```
knn<-function(x0,x,y,k){
  x=as.matrix(x)
  p=dim(x)[2]
  n=dim(x)[1]
  dis=rep(0,n)
  for(i in 1:p){
    dis= (x0[i]-x[,i])^2+dis
  }
  ind=order(dis)[1:k]
  mean(y[ind])
}
```

```
x_train=as.matrix(train[(train[,1]>1 & train[,1]<4),])
y_train=as.matrix(x_train[,1])
x_train=x_train[,-1]
x_test=as.matrix(test[(test[,1]>1 & test[,1]<4),])
y_test=as.matrix(x_test[,1])
x_test=x_test[,-1]
```

```
###testing error for least square##
n_test=nrow(x_test)
y_test_hat=rep(0, n_test)
y_test_hat=bls(x_test, x_train, y_train)
```

```

test_error_bls=0
for( i in 1:n_test){
    test_error_bls=test_error_bls+(y_test[i]-y_test_hat[i])^2}
test_error_bls=test_error_bls/n_test
test_error_bls

###traing error for least square##
n_train=nrow(x_train)
y_train_hat=rep(0, n_train)
y_train_hat=bls(x_train, x_train, y_train)
train_error_bls=0
for( i in 1:n_train){
    train_error_bls=train_error_bls+(y_train[i]-y_train_hat[i])^2}
train_error_bls=train_error_bls/n_train
train_error_bls

###testing error for KNN##
n_test=nrow(x_test)
k=c(1,3,5,7,15)
y_test_hat=matrix(0, nrow=n_test, ncol=length(k))
test_error_knn=rep(0,length(k))
for(j in 1: length(k)){
    for(r in 1:n_test){
        y_test_hat[r,j]=knn(x_test[r,], x_train, y_train,k[j])}

for( i in 1:n_test){
    test_error_knn[j]=test_error_knn[j]+(y_test[i]-y_test_hat[i,j])^2}
test_error_knn[j]=test_error_knn[j]/n_test}
test_error_knn

###training error for KNN##
n_train=nrow(x_train)
k=c(1,3,5,7,15)
y_train_hat=matrix(0, nrow=n_train, ncol=length(k))
train_error_knn=rep(0,length(k))
for(j in 1: length(k)){
    for(r in 1:n_train){
        y_train_hat[r,j]=knn(x_train[r,], x_train, y_train,k[j])}

for( i in 1:n_train){
    train_error_knn[j]=train_error_knn[j]+(y_train[i]-y_train_hat[i,j])^2}
train_error_knn[j]=train_error_knn[j]/n_train}
train_error_knn

```