Exercise 3

Bias/variance analysis in regression

G. Bontempi

Question

Let us consider the dependency where the conditional distribution of $\mathbf{y}$ is

$$\mathbf{y} = \sin(2\pi x_1 x_2 x_3) + \mathbf{w}$$

and $\mathbf{w} \sim N(0, \sigma^2)$ with $\sigma = 0.25$. Suppose that $\mathbf{x} \in \mathbb{R}^3$ has a 3D normal distribution with an identity covariance matrix. The number of observed input/output samples is $N = 100$.

Consider the following families of learners:

- constant model returning always zero
- constant model $h(x) = \beta_0$
- linear model $h(x) = x^T \beta$
- K nearest neighbour for $K = 1, 3, 5, 7$ where the distance is Euclidean

Implement for each learner above a function

```
learner<-function(Xtr,Ytr,Xts){
  ####
  ## Xtr [N,n] input training set
  ## Ytr [N,1] output training set
  ## Xts [Nts,n] input test set
  return(Yhat)

}
```

which returns a vector $[N_{ts}, 1]$ of predictions for the given input test set.

By using Monte Carlo simulation ($S = 100$ runs) and by using a fixed-input test set of size $N_{ts} = 1000$

- compute the average squared bias of all the learners,
- compute the average variance of all the learners,
- check the relation between squared bias, variance, noise variance and MSE
- define what is the best learner in terms of MSE,
- discuss the results.

NOTA BENE: the use of the R command `lm` is NOT allowed.

Learners

```
zeroL<-function(Xtr,Ytr,Xts){
  Nts=NROW(Xts)
  Yhat=numeric(Nts)

}

constantL<-function(Xtr,Ytr,Xts){
  Nts=NROW(Xts)
  Yhat=numeric(Nts)+mean(Ytr)
}

linearL<-function(Xtr,Ytr,Xts){
  Nts=NROW(Xts)
  N=NROW(Xtr)
  XXtr=cbind(numeric(N)+1,Xtr)
  XXts=cbind(numeric(Nts)+1,Xts)
  betahat=solve(t(XXtr)%*%XXtr)%*%t(XXtr)%*%Ytr
  Yhat=XXts%*%betahat
}

knnL<-function(Xtr,Ytr,Xts,K=1){
  Nts=NROW(Xts)
  N=NROW(Xtr)
  Yhat=numeric(Nts)
  for (i in 1:Nts){

    Distance=apply((Xtr-array(1,c(N,1))%*%Xts[i,])^2,1,mean)
    iD=sort(Distance, decreasing=FALSE, index=TRUE)$ix[1:K]
    Yhat[i]=mean(Ytr[iD])
  }
  Yhat
}
```

Monte Carlo Simulation

```
N=100 ## number of samples
Nts=1000
n=3
S=100 ## number of MC trials
models=c("zero","const","lin","1NN","3NN","5NN","7NN")
sdw=0.25 ## stanard deviation of noise
M=length(models)
Xts=array(rnorm(Nts*n),c(Nts,n))

fts=sin(2*pi*Xts[,1]*Xts[,2]*Xts[,3])
YH=array(0,c(S,Nts,M))
Ytrue=NULL

for (s in 1:S){
  Yts=sin(2*pi*Xts[,1]*Xts[,2]*Xts[,3])+rnorm(N,sd=sdw)
```

```
  Xtr=array(rnorm(N*n),c(N,n))
  Ytr=sin(2*pi*Xtr[,1]*Xtr[,2]*Xtr[,3])+rnorm(N,sd=sdw)

  Yhats1=zeroL(Xtr,Ytr,Xts)
  YH[s,,1]=Yhats1

  Yhats2=constantL(Xtr,Ytr,Xts)
  YH[s,,2]=Yhats2

  Yhats3=linearL(Xtr,Ytr,Xts)
  YH[s,,3]=Yhats3

  Yhats4=knnL(Xtr,Ytr,Xts,K=1)
  YH[s,,4]=Yhats4

  Yhats5=knnL(Xtr,Ytr,Xts,K=3)
  YH[s,,5]=Yhats5

  Yhats6=knnL(Xtr,Ytr,Xts,K=5)
  YH[s,,6]=Yhats6

  Yhats7=knnL(Xtr,Ytr,Xts,K=7)
  YH[s,,7]=Yhats7


  Ytrue<-rbind(Ytrue,Yts)
  cat(".")
}
```

```
## ...................................................................................................
```

```
mYH=apply(YH,c(2,3),mean)
vYH=apply(YH,c(2,3),var)

SBiases=(apply((fts-mYH)^2,2,mean))
Variances=apply(vYH,2,mean)

MSE=numeric(M)
for (j in 1:M)
  MSE[j]=mean((Ytrue-YH[,,j])^2)

print(SBiases+Variances+sdw^2)
```

```
## [1] 0.4536319 0.4568861 0.4750905 0.7430748 0.5241774 0.4888524 0.4776619
```

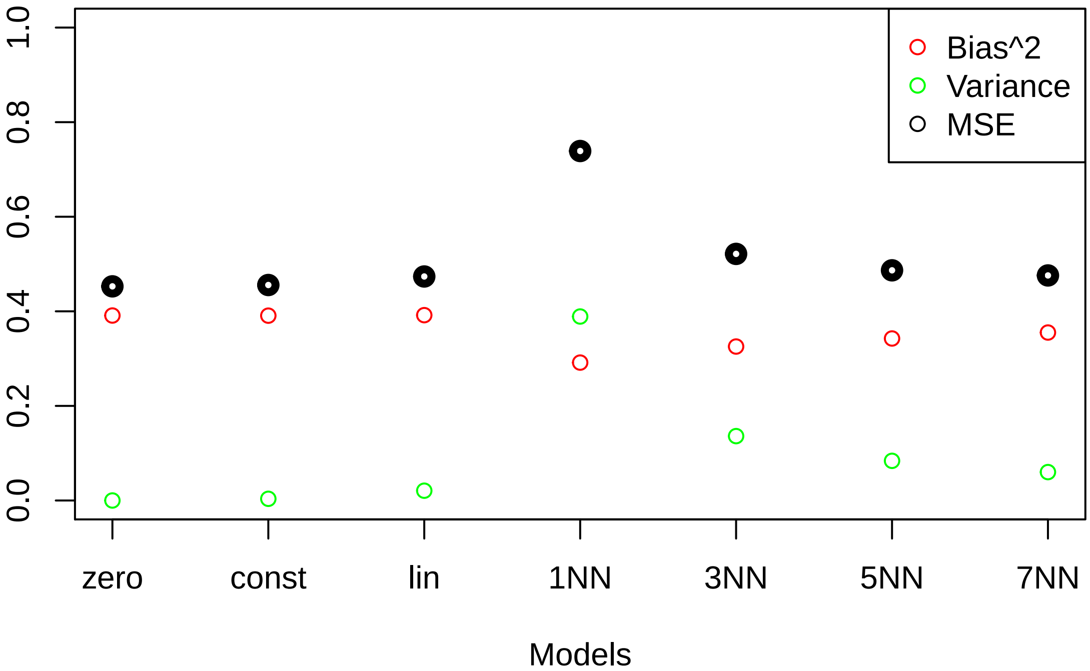
```
print(MSE)
```

```
## [1] 0.4527826 0.4556364 0.4736588 0.7389983 0.5214273 0.4867009 0.4758631
```

Here above we checked the identity between MSE, squared bias and variance

```
plot.default(factor(models, levels=models),SBiases,col="red",
             ylim=c(0,1),xaxt="n",
             xlab="Models",ylab="")
points(factor(models, levels=models),Variances,col="green")
points(factor(models, levels=models),MSE,col="black",lwd=4)
```

```r
axis(side=1, at=1:M,labels=models)
legend("topright",c("Bias^2","Variance","MSE"),pch=c(1,1,1),col=c("red","green","black"))
```




```r
bestModel=models[which.min(MSE)]
```

The plot shows that the first three learner have low variance but large bias. For the KNN learners it appears that the bias (variance) increases (decreases) by increasing K.

The best model in terms of MSE is **zero** since it shows the best tradeoff in terms of bias and variance. As you see it is not always the most sophisticated learning model which allows the best generalization!