Advanced Statistical Modelling: Ridge Regression

Ricard Meyerhofer & Joel Cantero

29/10/2019

Choosing the penalization parameter λ

The objective of this exercise is to implement Ridge Regression with two different approaches: $MSPE_{val}(\lambda)$ and $MSPE_{k-CV}(\lambda)$. In both cases we are going to take as input data the following:

- Matrix x and vector y corresponding to the training sample.
- Matrix x_{val} and vector y_{val} corresponding to the validation set.
- Vector *lambda.v* of candidate values for λ.

We are going to output for each element λ in *lambda.v* and the value of the $MSPE_{val}(\lambda)$ / $MSPE_{k-CV}(\lambda)$. Furthermore, we are going to plot these values against $log(1 + \lambda) - 1$.

Once we have build these two functions, we are going to use the prostate data used in class. We are going to choose a λ according to:

- Behaviour in the validation set (30 validations not included in the training sample)
- 5-fold, 10-fold cross-validation.
- Compare our results with those obtained with leave-one-out and generalized cross-validation.

Ridge regression based on $MSPE_{val}(\lambda)$

In order to choose the penalization parameter λ, we are going to write a function implementing the ridge regression penalization parameter λ choice based on the minimization of the $MSPE_{val}(\lambda)$.

```r
ridge <- function(x, y, x.val, y.val, lambda.v) {
  result <- data.frame(lambda=lambda.v, mspe=rep(0,length(lambda.v)),
                       df=rep(0,length(lambda.v)))
  x.svd <- svd(x)
  for(i in 1:length(lambda.v)) {
    lambda <- lambda.v[i]
    d_inv <- diag(1/(x.svd$d*x.svd$d - lambda))
    xx_inv <- t( solve( t(x) %*% x + lambda*diag(1,ncol(x)) ))
    beta <- xx_inv %*% t(x) %*% y
    y.hat <- x.val %*% beta
    mspe <- sum((y.val - y.hat)^2) / length(y.val)
    df <- sum(x.svd$d^2 / (x.svd$d^2 +lambda))
    result$mspe[i] <- mspe
    result$df[i] <- df
  }
  plot(mspe~log(1+lambda), result, col=2)
  lambda.min <- result$lambda[which.min(result$mspe)]
  abline(v=log(1+lambda.min),col=2,lty=2)
  plot(mspe~df, result, col=3)
  df.min <- result$df[which.min(result$mspe)]
  abline(v=df.min,col=3,lty=2)
  return(result)
}
```

Ridge regression based on $MSPE_{k-CV}(\lambda)$

Now, we will write an R function implementing the ridge regression penalization parameterλ choice based on k-fold cross-validation $MSPE_{kCV}(\lambda)$.

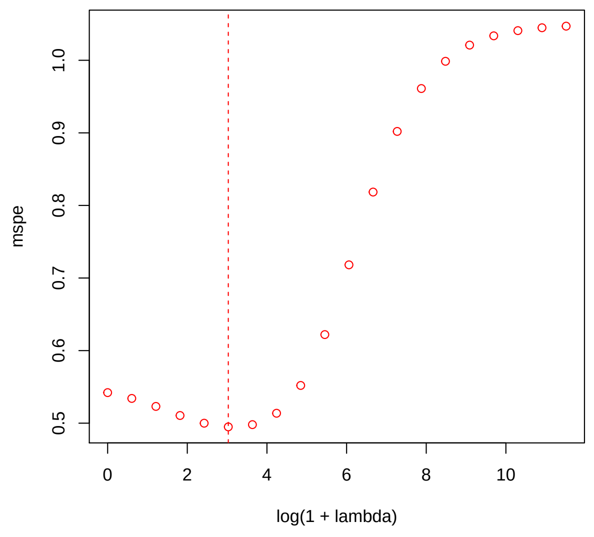
```
ridge_cv <- function(x, y, lambda.v, cv=10) {
  result <- data.frame(lambda=lambda.v, mspe=rep(0,length(lambda.v)),
                       df=rep(0,length(lambda.v)))
  for(i in 1:length(lambda.v)) {
    lambda.v[i] <- lambda.v[i]
    folds <- createFolds(1:nrow(x), k = cv)
    result.cv <- data.frame(mspe=rep(0,cv), df=rep(0,cv))
    for(j in 1:cv) {
      fold <- folds[[j]]
      x.train <- x[-fold,]
      y.train <- y[-fold]
      x.val <- x[fold,]
      y.val <- y[fold]
      x.svd <- svd(x.train)
      d <- x.svd$d
      v <- x.svd$v
      d_inv <- diag(1/(d*d - lambda.v[i]))
      xx_inv <- t( solve( t(x.train) %*% x.train + lambda.v[i]*diag(1,ncol(x)) ))
      beta <- xx_inv %*% t(x.train) %*% y.train
      y.hat <- x.val %*% beta
      mspe <- sum((y.val - y.hat)^2) / length(y.val)
      df <- sum(d^2 / (d^2 +lambda.v[i]))
      result.cv$mspe[j] <- mspe
      result.cv$df[j] <- df
    }
    result$mspe[i] <- mean(result.cv$mspe)
    result$df[i] <- mean(result.cv$df)

  }
  plot(mspe~log(1+lambda), result, col=2)
  lambda.min <- result$lambda[which.min(result$mspe)]
  abline(v=log(1+lambda.min),col=2,lty=2)
  plot(mspe~df, result, col=3)
  df.min <- result$df[which.min(result$mspe)]
  abline(v=df.min,col=3,lty=2)

  return(result)

}
```

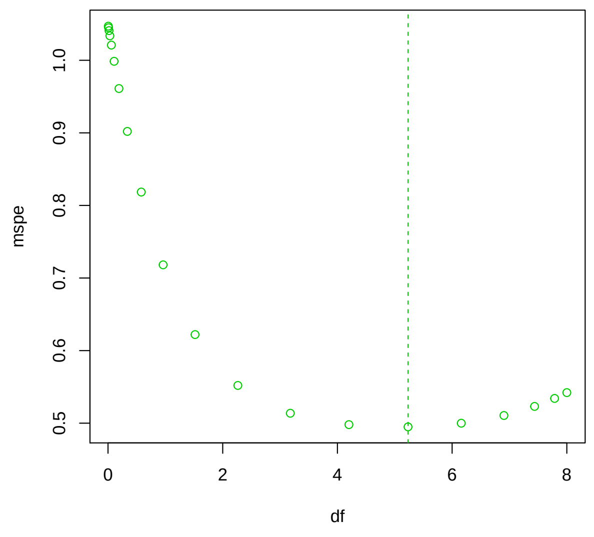
Comparation between penalization parameters of $MSPE_{val}(\lambda)$ and $MSPE_{k-CV}(\lambda)$

```
ridge_loocv_gcv <- function(x, y, lambda.v) {
  l <- length(lambda.v)
  result <- data.frame(lambda=lambda.v, loocv=rep(0,l), gcv=rep(0,l),df=rep(0,l))
  n <- nrow(x)
  x.svd <- svd(x)
  d <- x.svd$d
  v <- x.svd$v
  u <- x.svd$u
  for(i in 1:l) {
    lambda <- lambda.v[i]
    d_inv <- diag(1/(d^2 - lambda))
    xx_inv <- t( solve( t(x) %*% x + lambda*diag(1,ncol(x)) ))
    beta <- (xx_inv %*% t(x)) %*% y
    y.hat <- x %*% beta
    df <- sum(d^2 / (d^2 +lambda))
    h <- x %*% xx_inv %*% t(x)
    result$loocv[i] <- sum( ( (y - y.hat)/(1 - diag(h)) )^2 ) / n
    result$gcv[i] <- sum( ( (y - y.hat)/(1 - df/n) )^2 ) / n
    result$df[i] <- df
  }
  plot(loocv~log(1+lambda), result, col=2)
  lambda.min <- result$lambda[which.min(result$loocv)]
  abline(v=log(1+lambda.min),col=2,lty=2)
  plot(loocv~df, result, col=3)
  df.min <- result$df[which.min(result$loocv)]
  abline(v=df.min,col=3,lty=2)
  plot(gcv~log(1+lambda), result, col=2)
  lambda.min <- result$lambda[which.min(result$gcv)]
  abline(v=log(1+lambda.min),col=2,lty=2)
  plot(gcv~df, result, col=3)
  df.min <- result$df[which.min(result$gcv)]
  abline(v=df.min,col=3,lty=2)
  return(result)
}
```

Validation

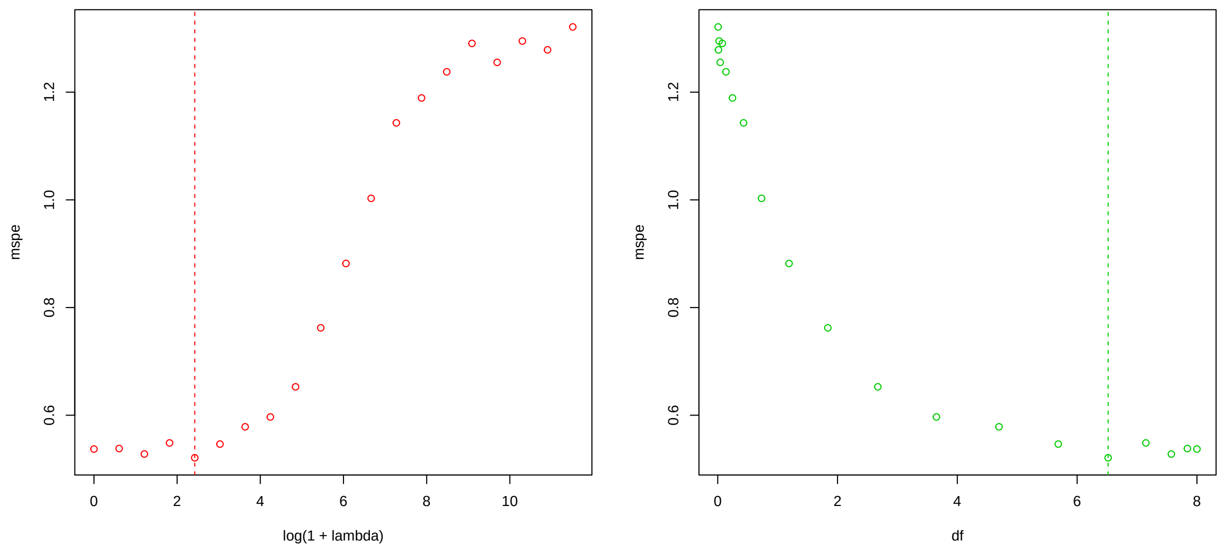
```
op<-par(mfrow=c(1,2))
result.valid <- ridge(x, y, x.val, y.val, lambda.v)
```

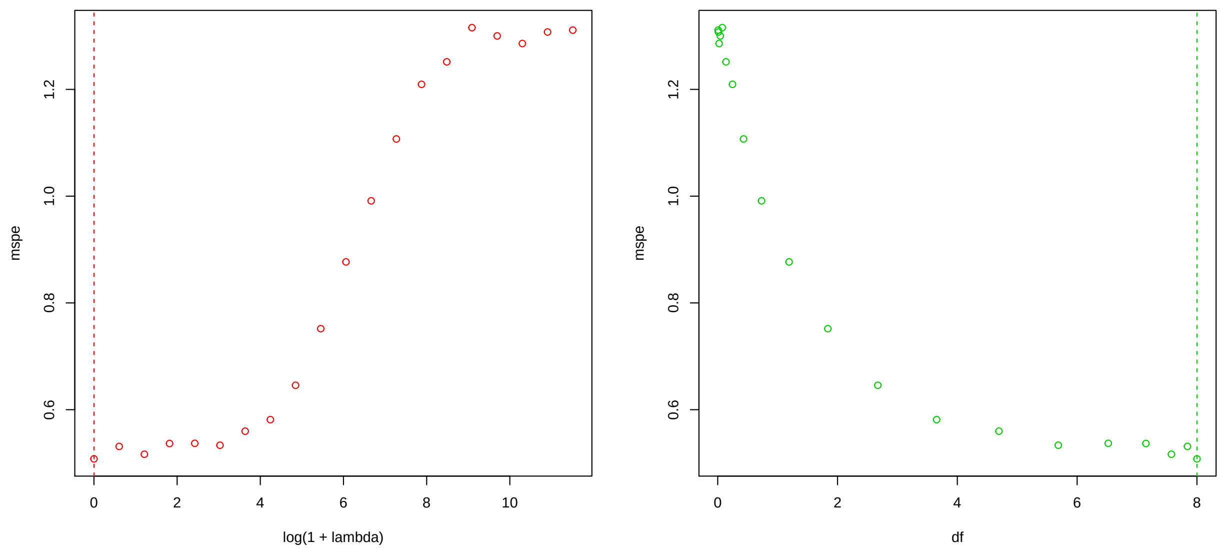




5-fold and 10-fold CV

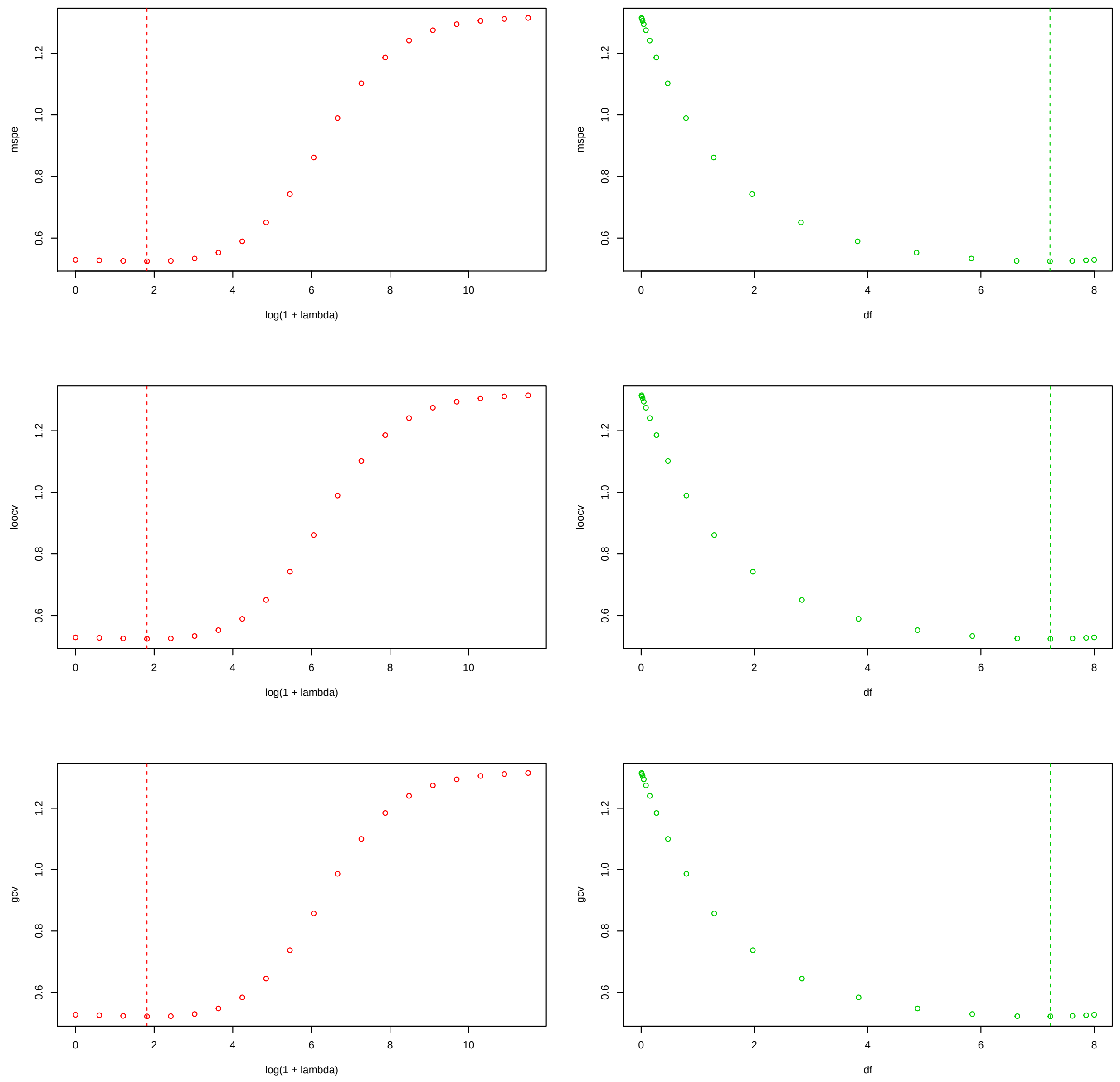
```
x <- scale(data[,1:8], center = T, scale = T)
y <- scale(data[,9], center = T, scale = F)
op<-par(mfrow=c(2,2))
result.5.cv <- ridge_cv(x, y, cv = 10, lambda.v)
result.10.cv <- ridge_cv(x, y, cv = 10, lambda.v)
```





LOOCV and GCV

```r
op<-par(mfrow=c(3,2))
result.loocv <- ridge_cv(x, y, cv = nrow(x), lambda.v) #n-fold CV
result.loocv.gcv <- ridge_loocv_gcv(x, y, lambda.v)
```



We can see from the plots that MSPE results are more solid in LOOCV and GCV than 5-fold and 10-fold. On the other hand, if we observe the validation data seems that it behaves the same way even that the curvature is different. We have to say that prostate dataset just has 97 observations and that has an impacts.

Ridge regression for the Boston Housing data

The Boston Housing dataset is a classical dataset which containes the values of 506 suburbs of Boston corresponding to 1978. This dataset can be found in many places but we are going to use a version with some corrections that was provided to us, which additionally includes the UTM coordinates of the geographical centers of each neighborhood. Therefore, the variables are the following:

Variable	Description	Type
CRIM	per capita crime rate by town	Numeric
ZN	proportion of residential land zoned for lots over 25,000 sq.ft.	Numeric
INDUS	proportion of non-retail business acres per town	Numeric
CHAS	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)	Factor
NOX	nitric oxides concentration (parts per 10 million)	Numeric
RM	average number of rooms per dwelling	Numeric
AGE	proportion of owner-occupied units built prior to 1940	Numeric
DIS	weighted distances to five Boston employment centres	Numeric
RAD	index of accessibility to radial highways	Numeric
TAX	full-value property-tax rate per $10,000	Numeric
PTRATIO	pupil-teacher ratio by town	Numeric
B	1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town	Numeric
LSTAT	% lower status of the population	Numeric
MEDV	Median value of owner-occupied homes in $1000's	Numeric

In this exercise we are going to use ridge regression on the Boston Housing dataset to fit the regression model where the response variable is $MEDV$ and the explanatory variables are the remaining 13 variables shown in the list. As we can see when loading the data, there are more variables than the ones listed ($TOWN$, $TOWNNO$, LON, LAT, $CMEDV$). We decided not to use them since the statement explicitly defines which to use.

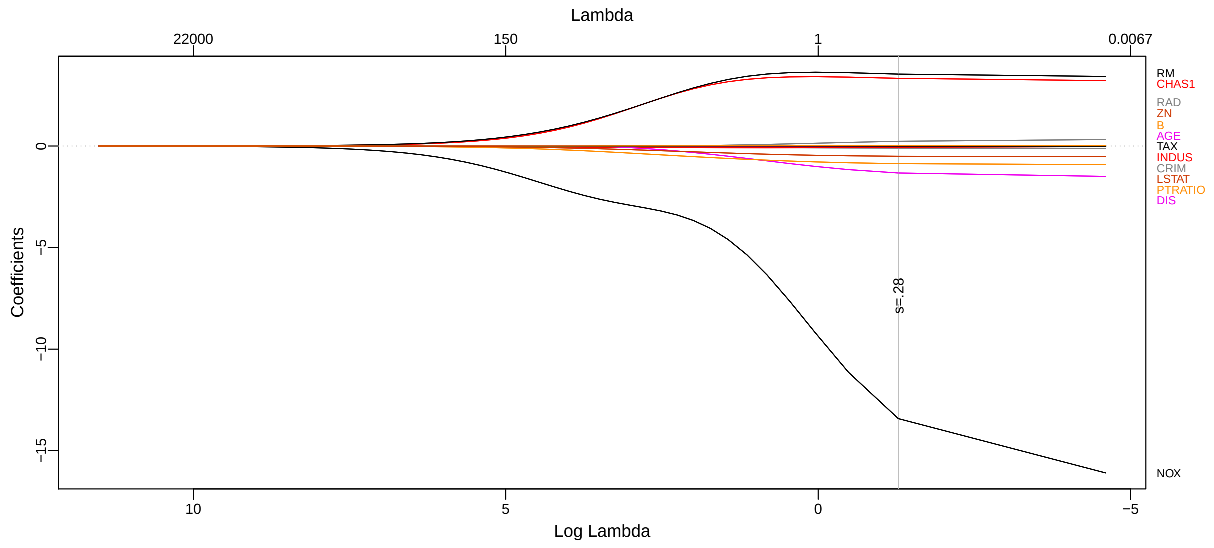
```
load("boston.Rdata")
names(boston.c)
dataset <- boston.c[,-c(1:5)] #"TOWN"    "TOWNNO"  "TRACT"   "LON"     "LAT"
dataset$CMEDV <- NULL
```

Besides from eliminating variables, we divided the dataset between train and test. To do so, we have decided to do 3/4 training, 1/4 test.

```
trainIndex <- createDataPartition(dataset$MEDV, p =0.75, list = F)
train <- dataset[trainIndex,]
test <- dataset[-trainIndex,]
```

We perform the ridge regression with 10 fold cross-validation in order to choose the tuning parameter lambda, we take the smallest lambda with *cv.out$lambda.min* and we use this lambda to make the predictions.

```
x <- model.matrix(MEDV~., train)[,-1]
y = train$MEDV
y.test = y[-trainIndex]
#perform cross-validation to choose tuning parameter lambda
lambda.max <- 1e5
n.lambdas <- 50
lambda.v <- exp(seq(10^-2,log(lambda.max+1),length=n.lambdas))-1
mod <- glmnet(x, y, alpha = 0, lambda = lambda.v)
ridge.mod <- cv.glmnet(x,y,alpha= 0, lambda = lambda.v)
bestlambda <- ridge.mod$lambda.min #lambda that results in lowest cross validation error
```



To select the best model, we now use 10x10-CV using the lambda that best minimised the error in cross-validation, which is lambda.ridge.

```
trainContrl <- trainControl (method="repeatedcv", number=10, repeats=10)
ridge.mod10 <- caret::train(MEDV ~ .,data = train,  trControl=trainContrl, method='glmnet',
                                    tuneGrid=expand.grid(alpha=0,lambda=bestlambda))

ridge <- glmnet(x, y, alpha = 0, lambda = bestlambda)
normalized <- (length(train$MEDV)-1)*var(train$MEDV)
NMSE.ridge.train.error <- crossprod(predict (ridge.mod10 ) - train$MEDV) / normalized
normTest <- (length(test$MEDV)-1)*var(test$MEDV)
sse_raw <- test$MEDV - ridge$a0- data.matrix(test[,-dim(train)[2]]) %*% ridge$beta
NMSE.ridge.test.error <- crossprod (as.matrix(sse_raw)) /normTest
```

The final model obtained has a $Df = 13$ which actually is the number of coefficients we had in our model and we have a deviance of 71.07% which has room for improvement but is not a bad result. Also the difference we have between train and test is not very big. We can take the following conclusions from the results:

- We can see that as we could expect by instinct, the price of a house increases as more rooms it has.
- If Charles River ($CHAR$) is near, the price of the suburb is higher.
- $LSTAT$ implies a decrease in the house of the price.
- RAD and $MEDV$ are directly related.
- Increase of $PTRATIO$ is associated with house price.
- An increase in DIS normally implies a decrease in $MEDV$.
- NOX puts the value down of a house.
- AGE and $INDUS$ doesn't seem to affect.

All the aformentioned makes sense since anyone wants to commute long ways to work, nor living in a polluted area, preferably everyone wants to be near nature, etc.