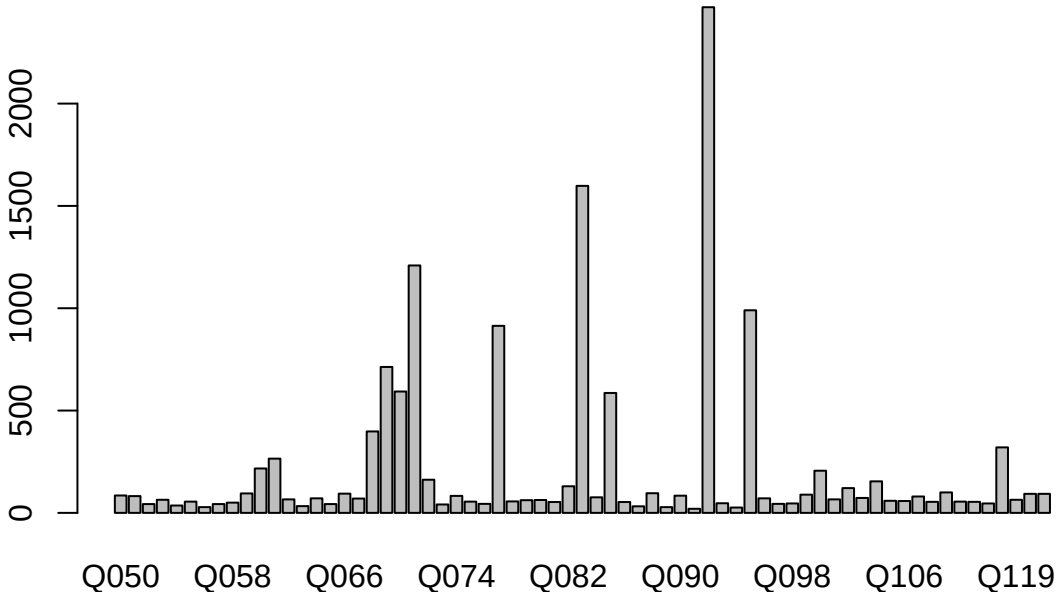
STAT 133 Final Report

Student: Xi Chen, SID: 22824137

1 Preliminary Data Analysis

First we analyze the if some questions are omitted more than others:

```
filtered.data = read.table("ling-data-clean.data", header=T)
vars = grep("Q", colnames(filtered.data), value=T)
omitted.count = apply(filtered.data[, vars] == 0, sum, MARGIN=c(2)) # filtered.data is raw data with sp
barplot(omitted.count)
```

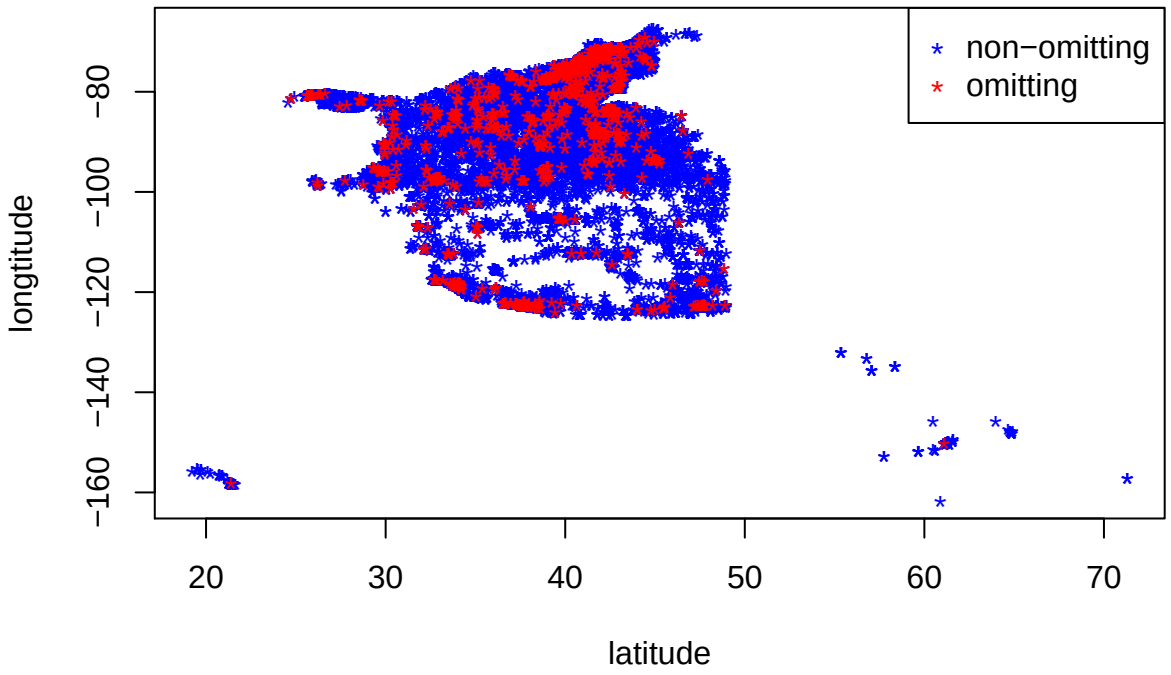


It's evident from the bar plot that certain questions are omitted significantly more than others. Next we will analyze what are the outliers and if the exclusion is related to geography:

```
outliers = names(omitted.count[omitted.count > quantile(omitted.count, probs=c(0.95))])
print(outliers)
```

```
## [1] "Q071" "Q083" "Q092" "Q095"
```

```
omitters.ind = filtered.data[, outliers] == 0
omitters = filtered.data[omitters.ind, ]
plot(filtered.data[-omitters.ind, "lat"], filtered.data[-omitters.ind, "long"], col="blue", xlab="latitu
points(omitters[, "lat"], omitters[, "long"], col="red", pch="*")
legend("topright", c("non-omitting", "omitting"), pch=c("*"), col=c("blue", "red"))
```

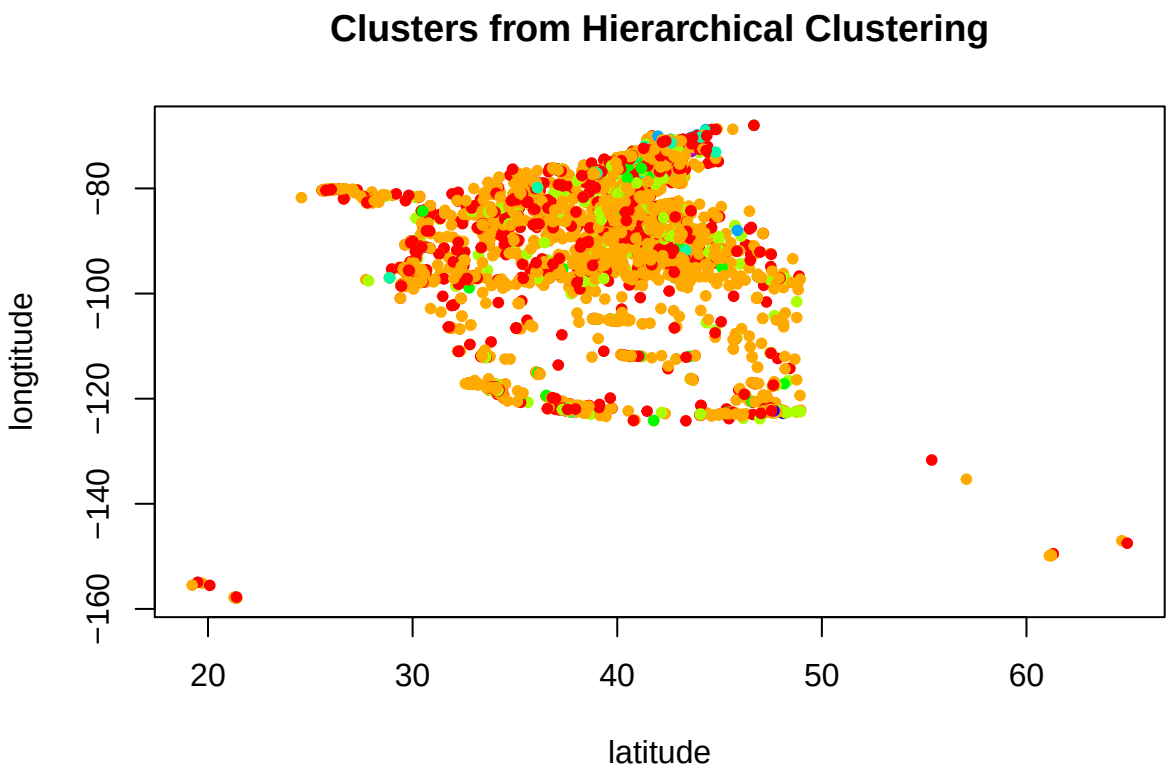


We can see from the graph that although there is a slight correlation between omitting these questions and geographic location, the connection is not conclusive.

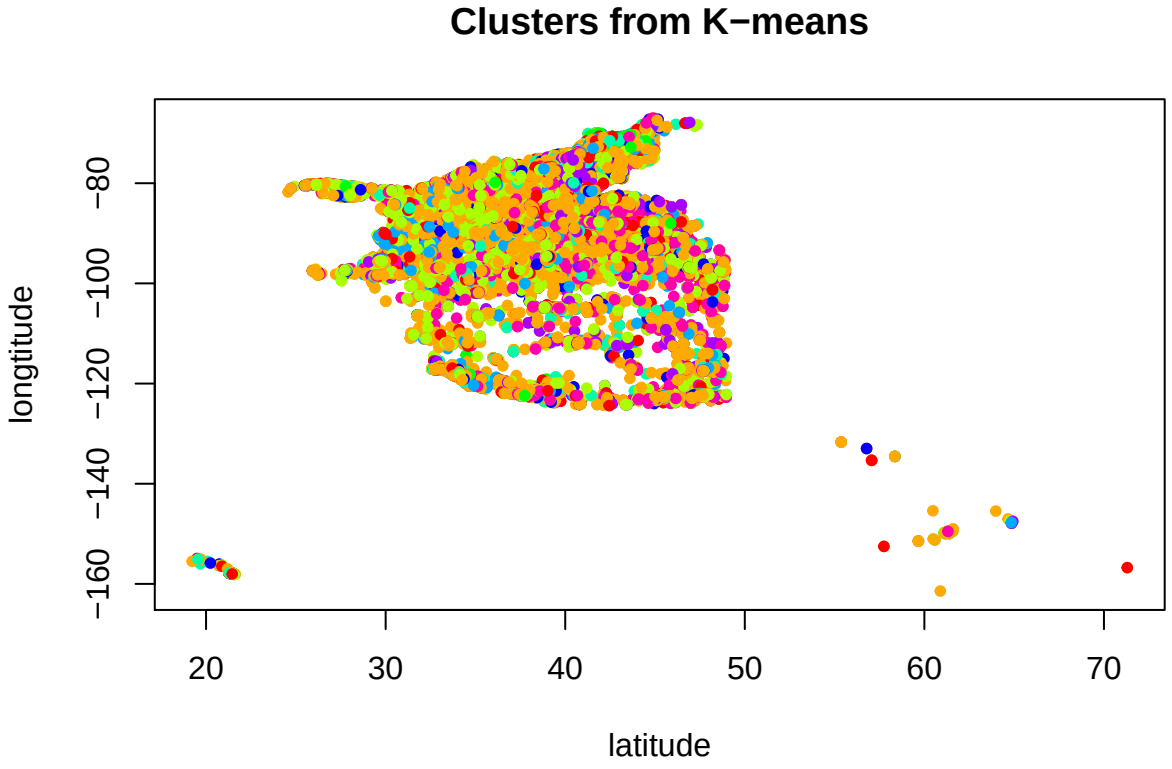
2 Relations between questions and geographic location

We will first look at the relations between some of the questions and location by inspecting the results of Hierarchical Clustering and K-means Clustering

```
binary.data = read.table("binary-ling-data.data", header=T)
# since hclust has a running time much worse than kmeans, we use a sampled set here to explore
sampled = sample(x=1:nrow(binary.data), size=5966)
binary.data.sampled = binary.data[sampled,]
binary.vars = grep("Q", names(binary.data), value=T)
selected.vars = binary.vars[65:107] # Q60-64
binary.dist = dist(binary.data.sampled[, selected.vars])
tree = hclust(binary.dist)
hclust.labels = cutree(tree, h=3)
colors = rainbow(max(hclust.labels))# brewer.pal(name='Blues',n=max(hclust.labels))
plot(binary.data.sampled[, "lat"], binary.data.sampled[, "long"], col=colors[hclust.labels], xlab="lati
```




```
kmeans.labels = kmeans(binary.data[, selected.vars], centers=max(hclust.labels))$cluster
plot(binary.data[, "lat"], binary.data[, "long"], col=colors[kmeans.labels], xlab="latitude", ylab="long
```



From both the results of K-means and Hierarchical Clustering, we can see that there are dominant clusters that are distributed relatively uniformly in terms of geographic-location. On the other hand, there exist

a certain number of smaller groups that exihbit stronger location association. This phenomenon can be interpreted as certain dialects are prevalent everywhere and hence do not indicate strong association with specific areas. On the other hand, some dialects might be tied more tightly to the locations that use the dialects.

Then we will investigate if we can use some questions to predict to answers of others. If the answer is yes, then it would be safe to assume that these questions do characterize different dialects. In particular, we will try to use a subset of the binarified data to predict another subset using multinomial logistic classification

```
train = sample(1:nrow(binary.data), size=5000)
test = sample(setdiff(1:nrow(binary.data), train), size=5000)

predicted.vars = vars[1:11] # Q50-Q60
temp = cbind(filtered.data[train, predicted.vars], binary.data[train, selected.vars])
library("nnet")
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
for (q in predicted.vars) {
  sink(file="/dev/null")
  multi = multinom(paste(q, " ~ ", paste(selected.vars, collapse="+")), data = temp, maxit=500)
  sink()
  predicted = predict(multi, binary.data[test, selected.vars])
  actual = filtered.data[test, q]
  print(paste("Using answers from Q60-64 to classify answer of ", q))
  print("Rate of naive classifier:")
  print(sum(Mode(actual) == actual)/ length(test))
  print("Rate of this classifier:")
  print(sum(predicted == actual) / length(test))
}
```

```
## [1] "Using answers from Q60-64 to classify answer of  Q050"
## [1] "Rate of naive classifier:"
## [1] 0.4124
## [1] "Rate of this classifier:"
## [1] 0.4328
## [1] "Using answers from Q60-64 to classify answer of  Q051"
## [1] "Rate of naive classifier:"
## [1] 0.6798
## [1] "Rate of this classifier:"
## [1] 0.6764
## [1] "Using answers from Q60-64 to classify answer of  Q052"
## [1] "Rate of naive classifier:"
## [1] 0.396
## [1] "Rate of this classifier:"
## [1] 0.396
## [1] "Using answers from Q60-64 to classify answer of  Q053"
## [1] "Rate of naive classifier:"
## [1] 0.8766
## [1] "Rate of this classifier:"
## [1] 0.8754
## [1] "Using answers from Q60-64 to classify answer of  Q054"
## [1] "Rate of naive classifier:"
```

```
## [1] 0.9146
## [1] "Rate of this classifier:"
## [1] 0.9146
## [1] "Using answers from Q60-64 to classify answer of  Q055"
## [1] "Rate of naive classifier:"
## [1] 0.8802
## [1] "Rate of this classifier:"
## [1] 0.8802
## [1] "Using answers from Q60-64 to classify answer of  Q056"
## [1] "Rate of naive classifier:"
## [1] 0.6162
## [1] "Rate of this classifier:"
## [1] 0.6272
## [1] "Using answers from Q60-64 to classify answer of  Q057"
## [1] "Rate of naive classifier:"
## [1] 0.69
## [1] "Rate of this classifier:"
## [1] 0.6892
## [1] "Using answers from Q60-64 to classify answer of  Q058"
## [1] "Rate of naive classifier:"
## [1] 0.5434
## [1] "Rate of this classifier:"
## [1] 0.5576
## [1] "Using answers from Q60-64 to classify answer of  Q059"
## [1] "Rate of naive classifier:"
## [1] 0.5006
## [1] "Rate of this classifier:"
## [1] 0.4966
## [1] "Using answers from Q60-64 to classify answer of  Q060"
## [1] "Rate of naive classifier:"
## [1] 0.6326
## [1] "Rate of this classifier:"
## [1] 1
```

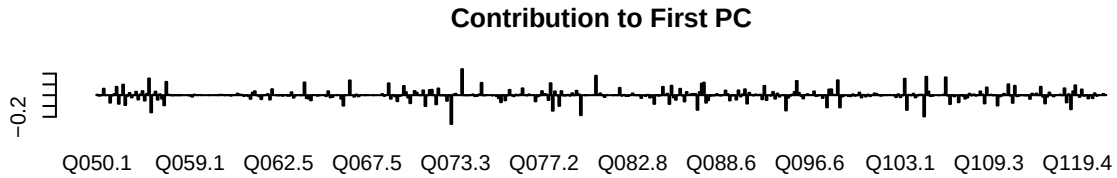
We will use the constant function $f(x) = Mode(X)$ as a baseline naive classifier to compare the multinomial classifier's performance. The fact that these naive classifiers can sometimes outperform naive classifiers suggest a certain level of connection between some of these questions, but it also revelas that there might not be associations between certain pairs of questions.

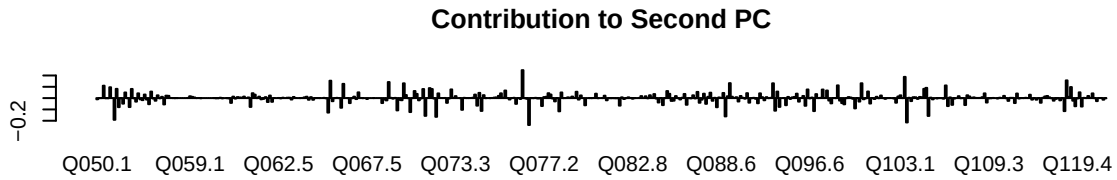
3 Dimensionality Reduction

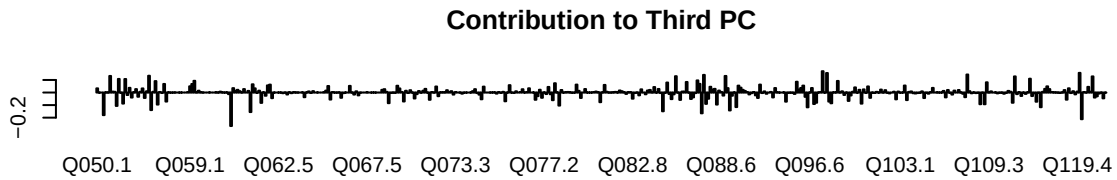
Since it's hard to explore high-dimensional data, we will apply the usual dimensionality reduction technique to aid our exploration. Notice here it doesn't make sense to apply PCA to the raw responses data, because the discrete values in each dimension (responses to questions) do not admit proper interpretation in real number domain.

```
# again, we use a sampled set instead of the whole data set
pca = prcomp(binary.data[, binary.vars])
# look at summary
# summary(pca)
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5    PC6    PC7
```

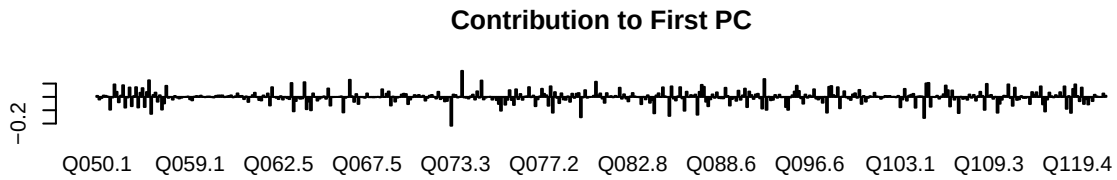
```
## Standard deviation     1.2191 1.1252 1.0225 0.8773 0.8460 0.8111 0.7855
## Proportion of Variance 0.0411 0.0350 0.0289 0.0213 0.0198 0.0182 0.0171
## Cumulative Proportion  0.0411 0.0762 0.1051 0.1265 0.1463 0.1645 0.1816
trans = solve(pca$rotation)
par(mfrow=c(3,1))
barplot(trans["PC1", ], main="Contribution to First PC")
barplot(trans["PC2", ], main="Contribution to Second PC")
barplot(trans["PC3", ], main="Contribution to Third PC")
```

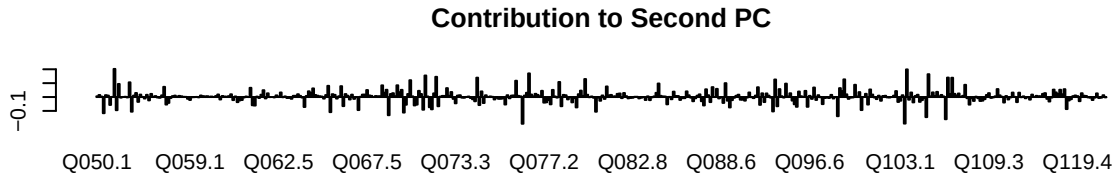


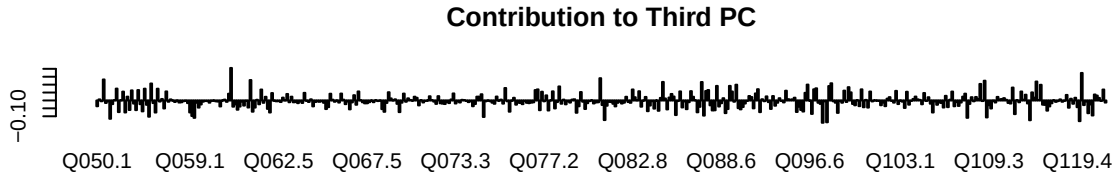





```
pca.scaled = prcomp(binary.data[, binary.vars], scale.=T)
# look at summary
# summary(pca.scaled)
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5    PC6     PC7
## Standard deviation     3.0254 2.7574 2.5768 2.3815 2.2235 2.1634 2.08508
## Proportion of Variance 0.0196 0.0163 0.0142 0.0121 0.0106 0.0100 0.00929
## Cumulative Proportion  0.0196 0.0358 0.0500 0.0621 0.0727 0.0827 0.09197
trans = solve(pca.scaled$rotation)
par(mfrow=c(3,1))
barplot(trans["PC1", ], main="Contribution to First PC")
barplot(trans["PC2", ], main="Contribution to Second PC")
barplot(trans["PC3", ], main="Contribution to Third PC")
```







We can observe that scaling the variables has a considerable effect on the results of PCA. This is expected since some of the dimensions contain extremely sparse data which make the absolute variance of that dimension smaller in comparison with others. Hence after scaling we can see that more variables have significant roles in the important principle components.

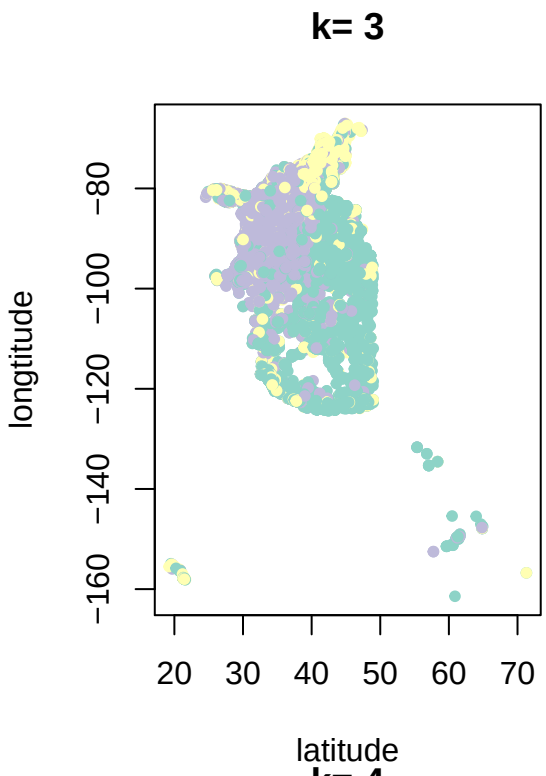
```
contrib = apply(abs(trans)[1:4, ], FUN=sum, MARGIN=c(2))
ind = which(contrib >= quantile(contrib, probs=c(0.99)))
significant.vars = binary.vars[ind]
```

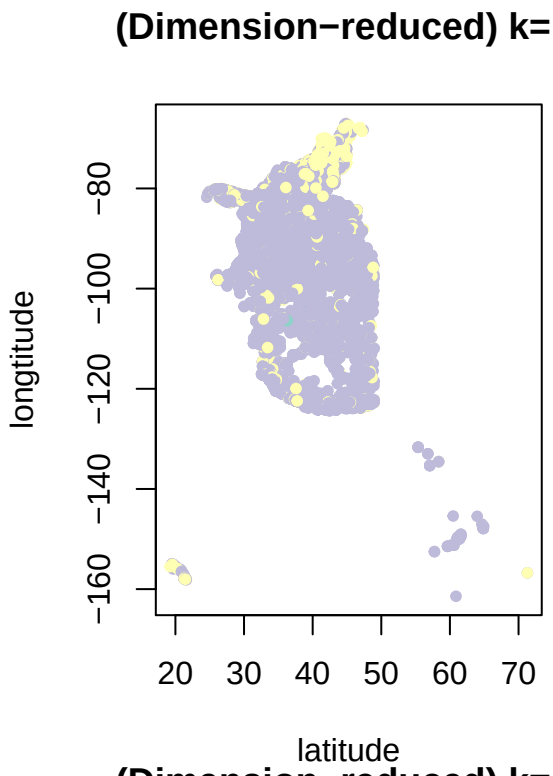
We can see that Questions 50, 86, 99, 105 and 115 are given more weights in principle components.

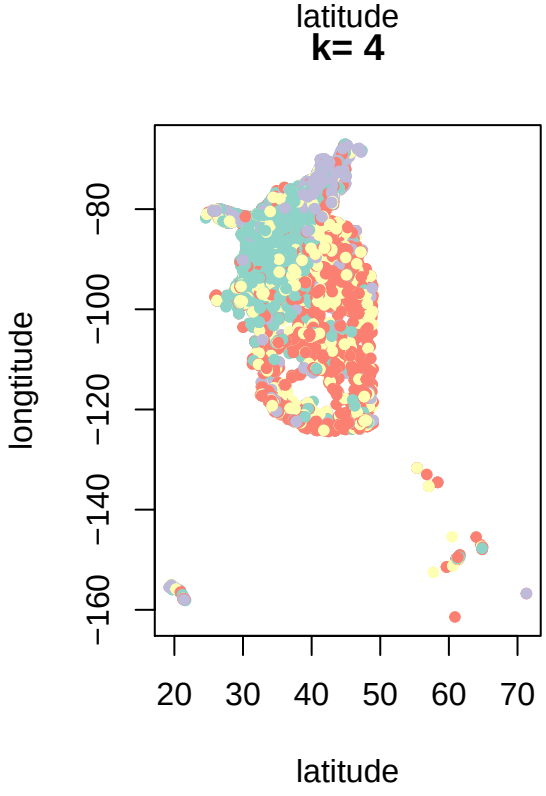
4 Clustering with Reduced-Dimension Data

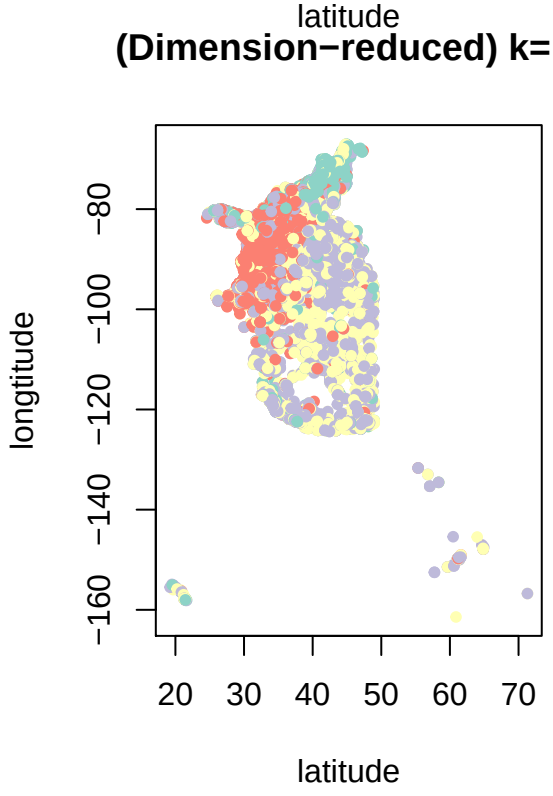
By choosing first few principle components from PCA, we can presumably preserve most intrinsic variation and get rid of noise. Hence it makes sense to revisit the problem of clustering with the reduced data. Here we will be using K-means to explore varying number of clusters.

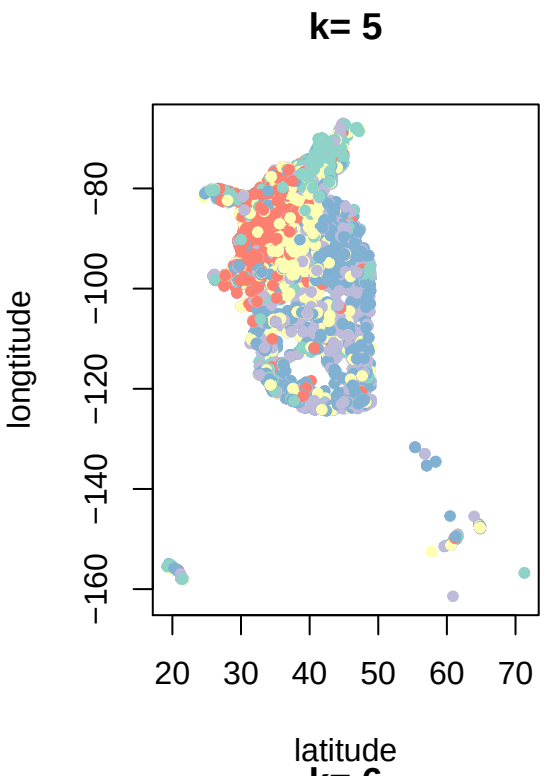
```
reduced = pca.scaled$x[, 1:217] # to account for 70% of the observed variance
par(mfrow=c(1,2))
library(RColorBrewer)
for (k in 3:7) {
  colors = brewer.pal(k,"Set3")
  clusters = kmeans(binary.data[, binary.vars], centers=k)$cluster
  plot(binary.data[, "lat"], binary.data[, "long"], col=colors[clusters], xlab="latitude", ylab="longti
  reduced.clusters = kmeans(reduced, centers=k)$cluster
  plot(binary.data[, "lat"], binary.data[, "long"], col=colors[reduced.clusters], xlab="latitude", ylab=
}
```

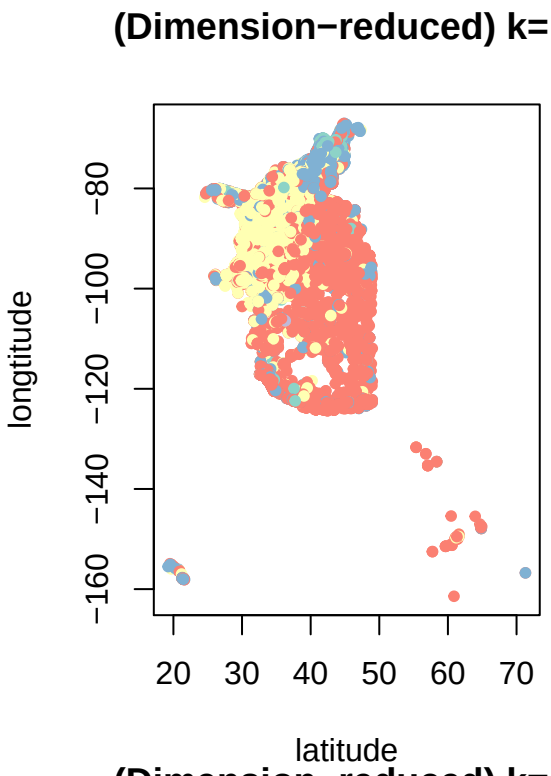

k= 3
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude

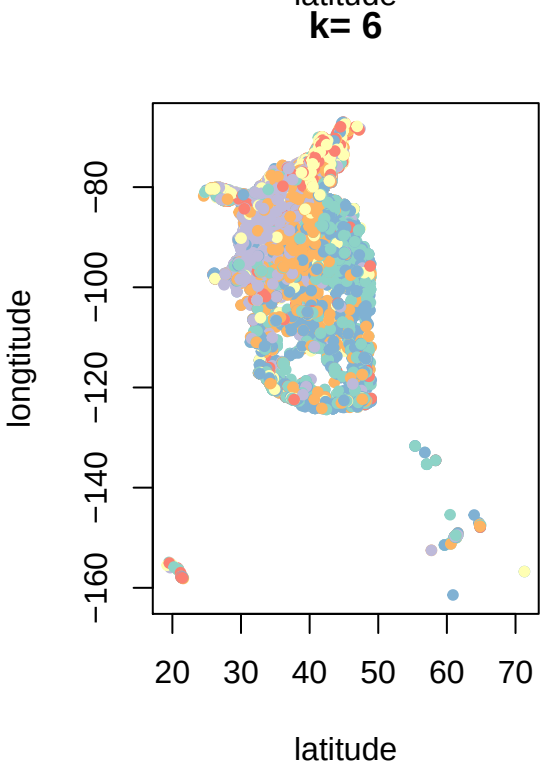

(Dimension−reduced) k= 3
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude

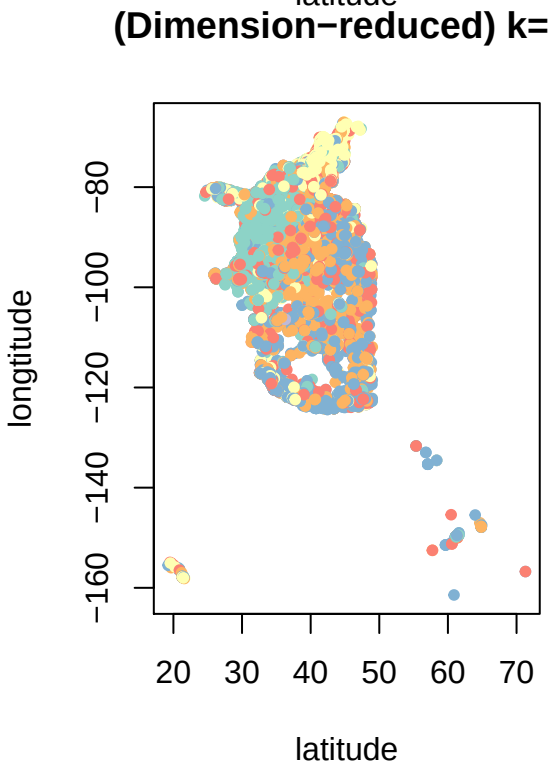

k= 4
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude

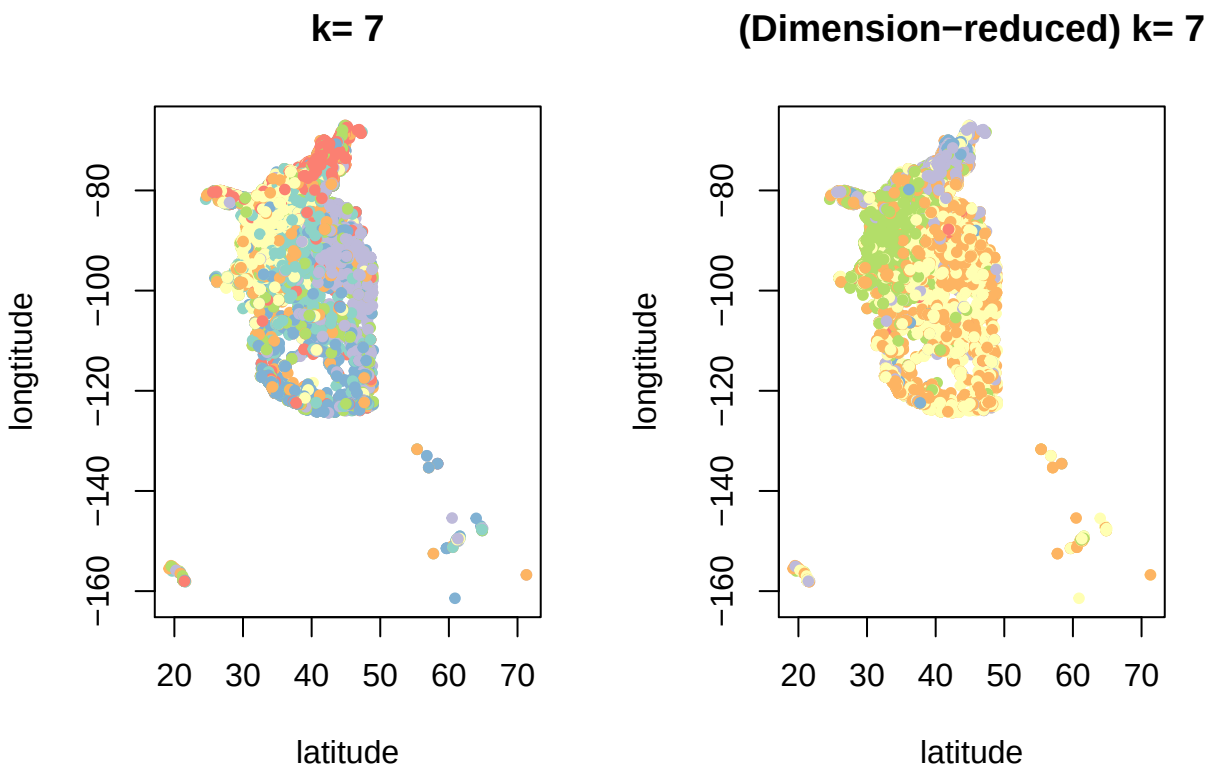

(Dimension−reduced) k= 4
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude


k= 5
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude


(Dimension−reduced) k= 5
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude


k= 6
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude

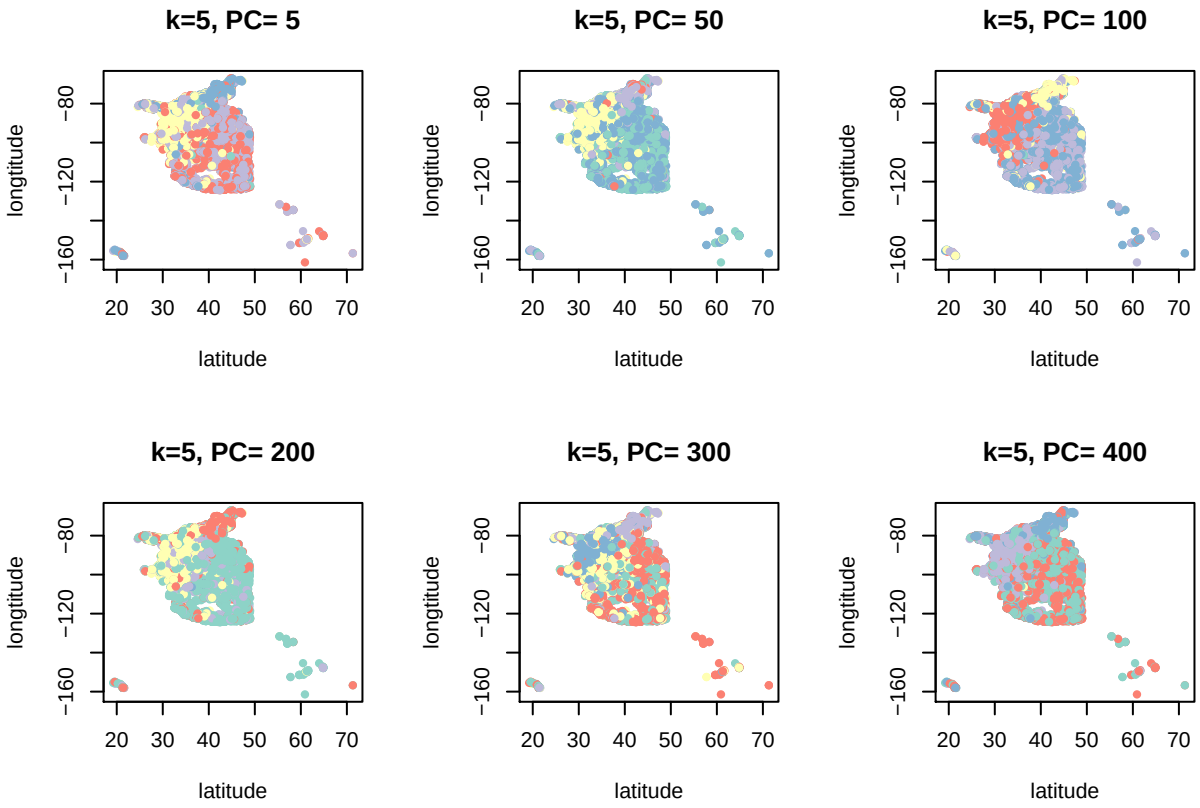

(Dimension−reduced) k= 6
longtitude
−80
−100
−120
−140
−160
20
30
40
50
60
70
latitude



We can see that clusters are relatively stable between the learnt from raw data and that from reduced data, except that in some cases the reduced data could yield results that merge some distinct clusters. On the other hand, as the number of clusters increases, the separability in terms of geo-location beomes weaker, which suggests that either these are dialects that are mixed in terms of geo-location or that these clusters that overlap significantly represent indeed geographically separate dialects but the survey questions cannot discern the differences between them.

In the end, we will investigate when we vary the number of principle components used, how stable is the clustering

```
par(mfrow=c(2,3))
for (i in c(5, 50, 100, 200, 300, 400)) {
  reduced = pca.scaled$x[, 1:i]# to account for 70% of the observed variance
  k=5
library(RColorBrewer)
  colors = brewer.pal(k,"Set3")
  reduced.clusters = kmeans(reduced, centers=k)$cluster
  plot(binary.data[, "lat"], binary.data[, "long"], col=colors[reduced.clusters], xlab="latitude", ylab=
}
```



From the above graph, we can see that albeit preserving general shape, the clustering is not very stable as the number of principle components used changes. It's also remarkable that using just 5 components can uncover almost as much structure as using many components.