Part4 Dimensionality Reduction

Simon Kim

2022-10-08

Data setup

For PCA, I picked dataset used on Notebook1.

```
library(caret)
```

Data cleaning

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(readr)
df <- read_csv("CASP.csv")
```

```
## Rows: 45730 Columns: 10

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (10): RMSD, F1, F2, F3, F4, F5, F6, F7, F8, F9
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df = subset(df, select = -c(RMSD))
df <- df[,1:9]

set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

PCA

run PCA of the data. Since we use F2~F9 as predictors, we will run PCA on those values.

```
pca_out <- preProcess(train[,2:9], method=c("center", "scale", "pca"))
pca_out
```

```
## Created from 36584 samples and 8 variables
##
## Pre-processing:
##   - centered (8)
##   - ignored (0)
##   - principal component signal extraction (8)
```

```
##   - scaled (8)
##
## PCA needed 4 components to capture 95 percent of the variance
```

deviding train and test value.

```
train_pc <- predict(pca_out, train[,1:9])
test_pc <- predict(pca_out, test[,])

summary(train_pc)
```

```
##        F1              PC1                PC2                 PC3
##  Min.   : 2392   Min.   :-4.1275   Min.   :-4.63160   Min.   :-47.13364
##  1st Qu.: 6939   1st Qu.:-1.6732   1st Qu.:-0.69410   1st Qu.: -0.21488
##  Median : 8896   Median :-0.5958   Median : 0.06206   Median : -0.01837
##  Mean   : 9873   Mean   : 0.0000   Mean   : 0.00000   Mean   :  0.00000
##  3rd Qu.:12136   3rd Qu.: 1.3262   3rd Qu.: 0.75827   3rd Qu.:  0.24568
##  Max.   :40035   Max.   :17.2017   Max.   : 5.06025   Max.   :  2.78392
##       PC4
##  Min.   :-2.33887
##  1st Qu.:-0.38242
##  Median :-0.04368
##  Mean   : 0.00000
##  3rd Qu.: 0.39427
##  Max.   :17.31966
```

```
summary(test_pc)
```

```
##        F1              PC1                  PC2                 PC3
##  Min.   : 2791   Min.   :-4.055640   Min.   :-4.32912   Min.   :-47.44577
##  1st Qu.: 6922   1st Qu.:-1.670113   1st Qu.:-0.71799   1st Qu.: -0.21691
##  Median : 8911   Median :-0.590847   Median : 0.03685   Median : -0.01744
##  Mean   : 9865   Mean   : 0.003437   Mean   :-0.02529   Mean   : -0.01944
##  3rd Qu.:12094   3rd Qu.: 1.286336   3rd Qu.: 0.76104   3rd Qu.:  0.25113
##  Max.   :38042   Max.   :17.719629   Max.   : 4.38492   Max.   :  1.77561
##       PC4
##  Min.   :-2.30902
##  1st Qu.:-0.39028
##  Median :-0.03719
##  Mean   : 0.01250
##  3rd Qu.: 0.41447
##  Max.   :15.30092
```

Linear Regression

Simple linear regression

```
lm <- lm(PC1~., data=train_pc)
summary(lm)
```

```
##
## Call:
## lm(formula = PC1 ~ ., data = train_pc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8039 -0.1872 -0.0021  0.1998  2.8061
```

```
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -5.692e+00  5.267e-03 -1080.69   <2e-16 ***
## F1           5.765e-04  4.944e-07  1166.05   <2e-16 ***
## PC2         -1.593e-01  1.867e-03   -85.33   <2e-16 ***
## PC3         -7.789e-02  2.414e-03   -32.26   <2e-16 ***
## PC4          4.936e-01  2.916e-03   169.28   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3783 on 36579 degrees of freedom
## Multiple R-squared:  0.9738, Adjusted R-squared:  0.9738
## F-statistic: 3.399e+05 on 4 and 36579 DF,  p-value: < 2.2e-16
```

```r
pred_lm <- predict(lm, newdata=test_pc)
cor_lm <- cor(pred_lm, test$F1)
mse_lm <- mean((pred_lm-test$F1)^2)
rmse_lm <- sqrt(mse_lm)
print(paste('correlation:', cor_lm))
```

```
## [1] "correlation: 0.985354921818796"
```

```r
print(paste('mse:', mse_lm))
```

```
## [1] "mse: 113841993.42143"
```

```r
print(paste('rmse:', rmse_lm))
```

```
## [1] "rmse: 10669.6763503599"
```

kNN Regression

Perform kNN regression, from calculating the best k value.

```r
train_df <- data.frame(train_pc$PC1, train_pc$PC2,train_pc$PC3,train_pc$PC4, train$F1)
test_df <- data.frame(test_pc$PC1, test_pc$PC2,test_pc$PC3,test_pc$PC4, test_pc$F1)
summary(train_df)
```

```
##   train_pc.PC1      train_pc.PC2       train_pc.PC3        train_pc.PC4
##  Min.   :-4.1275   Min.   :-4.63160   Min.   :-47.13364   Min.   :-2.33887
##  1st Qu.:-1.6732   1st Qu.:-0.69410   1st Qu.: -0.21488   1st Qu.:-0.38242
##  Median :-0.5958   Median : 0.06206   Median : -0.01837   Median :-0.04368
##  Mean   : 0.0000   Mean   : 0.00000   Mean   :  0.00000   Mean   : 0.00000
##  3rd Qu.: 1.3262   3rd Qu.: 0.75827   3rd Qu.:  0.24568   3rd Qu.: 0.39427
##  Max.   :17.2017   Max.   : 5.06025   Max.   :  2.78392   Max.   :17.31966
##     train.F1
##  Min.   : 2392
##  1st Qu.: 6939
##  Median : 8896
##  Mean   : 9873
##  3rd Qu.:12136
##  Max.   :40035
```

```r
cor_k <- rep(0, 20)
mse_k <- rep(0, 20)
i <- 1
for (k in seq(1, 39, 2)){
```

```r
 fit_k <- knnreg(train_df[,1:4],train_df[,5],k=k)
 pred_k <- predict(fit_k, test_df[,1:4])
 cor_k[i] <- cor(pred_k, test_df$test_pc.F1)
 mse_k[i] <- mean((pred_k - test_df$test_pc.F1)^2)
 print(paste("k=", k, cor_k[i], mse_k[i]))
 i <- i + 1
}
```

```
## [1] "k= 1 0.985623875204012 474731.996377348"
## [1] "k= 3 0.989622773700525 342186.08446261"
## [1] "k= 5 0.989982884649717 330868.608125667"
## [1] "k= 7 0.990084782959701 328060.866745799"
## [1] "k= 9 0.990049230969599 329978.476979682"
## [1] "k= 11 0.989979791587501 332738.875982619"
## [1] "k= 13 0.989851260695069 337502.445398491"
## [1] "k= 15 0.989770159676762 340605.973283183"
## [1] "k= 17 0.989665374356749 344340.587617581"
## [1] "k= 19 0.989620481799249 346223.685631823"
## [1] "k= 21 0.989531748741021 349522.62940513"
## [1] "k= 23 0.989510364564396 350523.717503451"
## [1] "k= 25 0.989383505994524 354966.793409493"
## [1] "k= 27 0.989269242183346 358907.073853238"
## [1] "k= 29 0.989176901845147 361980.390674654"
## [1] "k= 31 0.989073115946721 365556.740214615"
## [1] "k= 33 0.988969612260295 369136.463761181"
## [1] "k= 35 0.988864744035179 372711.611007099"
## [1] "k= 37 0.988743225658121 376746.429704915"
## [1] "k= 39 0.988654712805717 379743.656590129"
```

```r
which.min(mse_k)
```

```
## [1] 4
```

```r
which.max(cor_k)
```

```
## [1] 4
```

therefore, the best k is 4

```r
fit <- knnreg(train_df[,1:4],train_df[,5],k=4)

pred_knn <- predict(fit, test_df[,1:4])
cor_knn <- cor(pred_knn, test_df$test_pc.F1)
mse_knn <- mean((pred_knn - test_df$test_pc.F1)^2)
rmse_knn <- sqrt(mse_knn)
print(paste('correlation:', cor_knn))
```

```
## [1] "correlation: 0.989780344940679"
```

```r
print(paste('mse:', mse_knn))
```

```
## [1] "mse: 337245.098519543"
```

```r
print(paste('rmse:', rmse_knn))
```

```
## [1] "rmse: 580.728076228059"
```

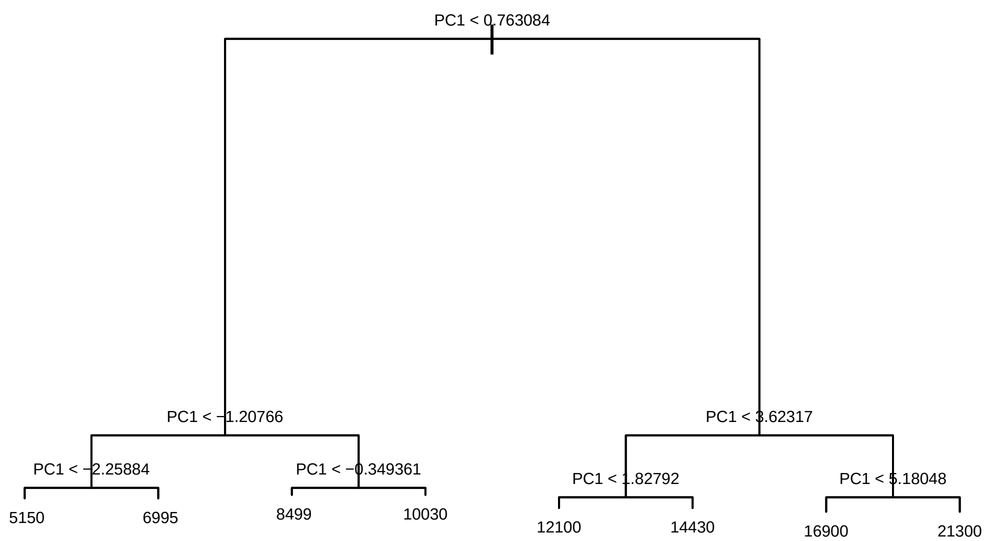
Decision Tree

Perform the decision tree regression model with all predictors of PCA train model. As we can see, the model is only actually use PC1 as the predictor. It is because tree use some indication such as entropy, information gain and Gini index to select a good information variables. This confirm the information we have when we exploring the data above.

```
library(tree)
colnames(train_df) <- c("PC1", "PC2", "PC3", "PC4", "F1")
colnames(test_df) <- c("PC1", "PC2", "PC3", "PC4", "F1" )
set.seed(1234)
tree1 <- tree(F1~., data=train_df)
summary(tree1)
```

```
##
## Regression tree:
## tree(formula = F1 ~ ., data = train_df)
## Variables actually used in tree construction:
## [1] "PC1"
## Number of terminal nodes:  8
## Residual mean deviance:  1170000 = 4.281e+10 / 36580
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean    3rd Qu.      Max.
## -16050.00   -661.60    -17.14      0.00     635.10  18740.00
```

```
plot(tree1)
text(tree1, cex=0.5, pretty=0)
```




```
pred_tree <- predict(tree1, newdata=test_df)
cor_tree <- cor(pred_tree, test_df$F1)
mse_tree <- mean((pred_tree - test_df$F1)^2)
rmse_tree <- sqrt(mse_tree)
print(paste('correlation:', cor_tree))
```

```
## [1] "correlation: 0.961515693338096"
```

```
print(paste('mse:', mse_tree))
```

```
## [1] "mse: 1248871.2343931"
```

```r
print(paste('rmse:', rmse_tree))
```

```
## [1] "rmse: 1117.52907541285"
```

Comparing the result

Model	Correlation	rmse
Linear Regression	0.998812915661555	198.129236023442
Linear Regression(PCA)	0.985354921818796	10669.6763503599
kNN Regression	0.998008972762981	256.639383652896
kNN Regression(PCA)	0.989780344940679	580.728076228059
Decision Tree Regression	0.981994447324154	769.473275456637
Decision Tree Regression(PCA)	0.961515693338096	1117.52907541285

As you can see from the Notebook1, the data is highly linear. It has the highest correlation and lowest rmse, which means highest accuracy among the 6 models. After PCA, the correlation lowered slightly, and rmse is higher. This means PCA didn't work properly for linear model, which is already linear for the certain attributes. PCA also affected KNN and Decision tree in the same way.
From the result above, We can see that PCA can affect the regression result in both good and bad direction. This time, the data set was already linearly correlated between certain attributes, and therefore dimensionality reduction made the result worse since it polymerize good attributes and bad attributes.

Linear Discriminant Analysis

Linear Discriminant Analysis, Also known as LDA is good for classification. Since the data we used for PCA doesn't have class, LDA will not have an advantage performing.

This time, i will perform LDA with employeeData from Notebook2

```r
library(MASS)

df2 <- read_csv("employeeData.csv")
```

```
## Rows: 4410 Columns: 24
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (8): Attrition, BusinessTravel, Department, EducationField, Gender, Job...
## dbl (16): Age, DistanceFromHome, Education, EmployeeCount, EmployeeID, JobLe...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
df2 <- na.omit(df2)

df2 <- df2[,c(1:7,9:15,17,19:23)]

df2$Attrition <- as.factor(df2$Attrition)
```

```r
set.seed(1234)
i <- sample(1:nrow(df2), 0.8*nrow(df2), replace=FALSE)
train2 <- df2[i,]
test2 <- df2[-i,]

lda1 <- lda(JobRole~., data=train2)
```

predict on test

```
lda_pred <- predict(lda1, newdata=test2, type="class")
```

plot on test

```
plot(lda_pred$x[,1], lda_pred$x[,2], pch=c(23,21,22)[unclass(lda_pred$class)], bg=c("red","green","blue
```