proj_section4

2023-03-25

proj_section4

2023-03-25 Preprocess data.

```
set.seed(0)
library(MASS)
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
## Warning in system("timedatectl", intern = TRUE): running command 'timedatectl'
## had status 1
```

```
library(compiler)
library(Metrics)
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

```
enableJIT(level = 3)
```

```
## [1] 3
```

```
df <- read.csv('unpopular_songs.csv')
df$explicit <- as.integer(factor(df$explicit))-1
split <- sample(1:length(df[,1]), length(df[,1])*0.8, replace=FALSE)
train <- df[split,]
test <- df[-split,]
```

KNNReg reproduction

```
knn <- knnreg(train[,1:13], train$popularity, k = 50)
preds <- predict(knn, test[,1:13])
print(paste("repro knn corr =", cor(preds, test$popularity)))
```

```
## [1] "repro knn corr = 0.131626601949771"
```

```
print(paste("repro knn mse =", mean((preds- test$popularity)^2)))
```

```
## [1] "repro knn mse = 15.7633791071049"
```

```
print(paste("repro knn acc =", accuracy(test$popularity, preds)))
```

```
## [1] "repro knn acc = 0.00183823529411764"
```

PCA

```
pca <- preProcess(train[,1:14], method=c('pca'))
pca
```

```
## Created from 8701 samples and 14 variables
##
## Pre-processing:
##   - centered (14)
##   - ignored (0)
##   - principal component signal extraction (14)
##   - scaled (14)
##
## PCA needed 12 components to capture 95 percent of the variance
```

```
pca_train <- predict(pca, train[,])
pca_test <- predict(pca, test[,])
df_pca_train <- data.frame(pca_train[,5:14], train$popularity)
df_pca_test <- data.frame(pca_test[,5:14], test$popularity)
```

KNN Regression through the PCA

```
fit <- knnreg(df_pca_train[,], train$popularity, k=50)
preds <- predict(fit, df_pca_test[,])
print(paste("mse =",mean((preds-test$popularity)^2)))
```

```
## [1] "mse = 0.0852228038176776"
```

```
print(paste("correlation =",cor(preds,test$popularity)))
```

```
## [1] "correlation = 0.997493693667381"
```

```
print(paste("acc =", accuracy(test$popularity, preds)))
```

```
## [1] "acc = 0.0229779411764706"
```

This seems to somehow increase the correlation and decrease the mse, so PCA has increased the accuracy of the model and not decreased it.

LDA

The dataframe is reduced due to the processing power required.

```
small_df <- df[1:1000,1:14]
split <- sample(1:length(small_df[,1]), length(small_df[,1])*0.8, replace=FALSE)
small_train <- small_df[split,]
small_test <- small_df[-split,]
lda <- lda(small_train$popularity~., data=small_train)
lda_pred <- predict(lda, newdata=small_test, type='class')
print(paste("correct =",mean(lda_pred$class==small_test$popularity)))
```

```
## [1] "correct = 0.295"
```

```
print(paste("acc =", accuracy(small_test$popularity, lda_pred$class)))
```

```
## [1] "acc = 0.295"
```

This also seems to have increased the accuracy from phase 1 in particular. I don't know why each of these increased the accuracy of each worked.