

# CS498AML\_HW1\_panz2

PanZhang

## Part 1A

```
accuracies
## [1] 0.7364865 0.6333333 0.6891892 0.7241379 0.7089552 0.6609195 0.6751592
## [8] 0.5987261 0.6709677 0.7304965
ave_accuracy=mean(accuracies)
ave_accuracy
## [1] 0.6828371
```

## Part 1B

```
accuracies
## [1] 0.6566265 0.6927711 0.6556291 0.7161290 0.6870748 0.6772152 0.6515152
## [8] 0.6397059 0.6274510 0.6993007
ave_accuracy=mean(accuracies)
ave_accuracy
## [1] 0.6703418
```

## Part 1D

```
accuracies
## [1] 0.8333333 0.7880795 0.7101449 0.7450980 0.8383234 0.7619048 0.7122302
## [8] 0.7727273 0.7352941 0.7484277
mean(accuracies)
## [1] 0.7645563
```

## Code:

```
naivebayes_update = function (datafr){
  #split data into train and test
  index=sample(2, nrow(datafr), replace=T, prob=c(0.8,0.2))
  data.train <- data[index==1, ]
  dim(data.train)
  data.test <- data[index==2, ]
  dim(data.test)
  #calculate posterior and class-conditional distributions
  labels=data.train[,ncol(datafr)]
  num_feature = ncol(datafr)-1
  label = unique(labels)
  feature = array(NA,dim=c(length(label),num_feature,2))
  priori = array(NA,rep(length(label)))
  for (i in 1:length(label)){
    label_class_sum = length(labels[labels==label[i]])
    priori[i] = label_class_sum/length(labels)
    attrs_condition = data.train[which(data.train$X1==label[i]),]
    for (j in 1:num_feature){
      attrs=attrs_condition[,j]
      #calculate mean and sd for each attribute
      feature[i,j,1]= mean(attrs,na.rm=TRUE)
```

```

        feature[i,j,2]= sd(attrs,na.rm=TRUE)
        #here we caculate mean and sd with NA removed.
    }
}
#apply mode to test data
right_class=0
for (k in 1:nrow(data.test)){
    p=array(NA,dim=c(length(label),num_feature))
    result=array(NA,rep(length(label)))
    for (i in 1:length(label)){
        for (j in 1:length(num_feature)){
            if (data.test[k,j]==FALSE){
                p[i][j]=1 }
            else{ p[i][j]=dnorm(data.test[k,j], feature[i,j,1],feature[i,j,2])}
            if (j==1){result[i]= p[i][j]}
            else{result[i]=result[i]*p[i][j]}
        }
        result[i]=result[i]*priori[i]
    }
    my_lable=label[which.max(result)]
    if (data.test[k,ncol(datafr)] == my_lable){
        right_class = right_class + 1
    }
}
}
accuracy=right_class/nrow(data.test)
return (accuracy)

```

```

}

SVMLight:
accuracies= array(NA,rep(10))
for (i in 1:10){
    index=sample(2, nrow(data_svm), replace=T, prob=c(0.8,0.2))
    data_svm.train <- data[index==1, ]
    dim(data_svm.train)
    data_svm.test <- data[index==2, ]
    dim(data_svm.test)
    svm = svmlight(data_svm.train[,1:8], factor(data_svm.train[,9]), pathsvm="/
Users/panzhang/Desktop/CS498AML/svm_light/")
    labels = predict(svm, data_svm.test[,1:8])
    answers = labels$class
    #accuracy
    correct <- sum(answers == data_svm.test[,9])
    accuracies[i] = correct / nrow(data_svm.test)
}
accuracies

```

x	Method	Accuracy
1	Gaussian + untouched	0.533
2	Gaussian + stretched	0.816
3	Bernoulli + untouched	0.837
4	Bernoulli + stretched	0.801
5	10 trees + 4 depth + untouched	0.718
6	10 trees + 4 depth + stretched	0.73
7	10 trees + 16 depth + untouched	0.9655
8	10 trees + 16 depth + stretched	0.972
9	30 trees + 4 depth + untouched	0.7605
10	30 trees + 4 depth + stretched	0.7545
11	30 trees + 16 depth + untouched	0.979
12	30 trees + 16 depth + stretched	0.9805

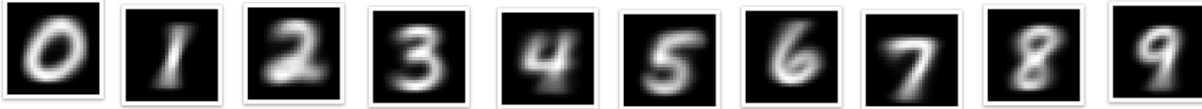
9 submissions for <a href="#">Pan Zhang</a>			Sort by	Most recent ▼
All	Successful	Selected		
Submission and Description		Public Score	Use for Final Score	
<a href="#">panz2_7.csv</a> a minute ago by <a href="#">Pan Zhang</a> <a href="#">add submission details</a>		0.95890	<input type="checkbox"/>	
<a href="#">panz2_9.csv</a> a minute ago by <a href="#">Pan Zhang</a> <a href="#">add submission details</a>		0.75405	<input type="checkbox"/>	
<a href="#">panz2_11.csv</a> 2 minutes ago by <a href="#">Pan Zhang</a> <a href="#">add submission details</a>		0.96900	<input type="checkbox"/>	
<a href="#">panz2_12.csv</a> 21 minutes ago by <a href="#">Pan Zhang</a> <a href="#">add submission details</a>		0.97265	<input type="checkbox"/>	
<a href="#">panz2_10.csv</a> 22 minutes ago by <a href="#">Pan Zhang</a> <a href="#">add submission details</a>		0.77560	<input type="checkbox"/>	
<a href="#">panz2_8.csv</a> 22 minutes ago by <a href="#">Pan Zhang</a> <a href="#">add submission details</a>		0.96460	<input type="checkbox"/>	



panz2\_1\_0.png panz2\_1\_1.png panz2\_1\_2.png panz2\_1\_3.png panz2\_1\_4.png panz2\_1\_5.png panz2\_1\_6.png panz2\_1\_7.png panz2\_1\_8.png panz2\_1\_9.png



panz2\_2\_0.png panz2\_2\_1.png panz2\_2\_2.png panz2\_2\_3.png panz2\_2\_4.png panz2\_2\_5.png panz2\_2\_6.png panz2\_2\_7.png panz2\_2\_8.png panz2\_2\_9.png



panz2\_3\_0.png panz2\_3\_1.png panz2\_3\_2.png panz2\_3\_3.png panz2\_3\_4.png panz2\_3\_5.png panz2\_3\_6.png panz2\_3\_7.png panz2\_3\_8.png panz2\_3\_9.png



panz2\_4\_0.png panz2\_4\_1.png panz2\_4\_2.png panz2\_4\_3.png panz2\_4\_4.png panz2\_4\_5.png panz2\_4\_6.png panz2\_4\_7.png panz2\_4\_8.png panz2\_4\_9.png

## Code:

```
import numpy as np
import scipy.misc
from pandas import Series, DataFrame
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import BernoulliNB
from sklearn.metrics import accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_extraction import DictVectorizer
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import string

#GaussianNB
model = GaussianNB()
model.fit(untouched_train_X, untouched_train_Y)
score=model.score(untouched_val_X,untouched_val_Y)
print(score)
predict = model.predict(untouched_test_X)

model.fit(stretched_train_X, stretched_train_Y)
score=model.score(stretched_val_X, stretched_val_Y)
print(score)
predict = model.predict(stretched_test_X)
```

```

#BernoulliNB
model = BernoulliNB()
model.fit(untouched_train_X, untouched_train_Y)
score=model.score(untouched_val_X,untouched_val_Y)
print(score)
predict = model.predict(untouched_test_X)

model.fit(stretched_train_X, stretched_train_Y)
score=model.score(stretched_val_X, stretched_val_Y)
print(score)
predict = model.predict(stretched_test_X)

#RandomForest
model= RandomForestClassifier(n_estimators=10, max_depth=4)
model.fit(untouched_train_X, untouched_train_Y)
score=model.score(untouched_val_X,untouched_val_Y)
print(score)
predict = model.predict(untouched_test_X)

model= RandomForestClassifier(n_estimators=10, max_depth=16)
model.fit(untouched_train_X, untouched_train_Y)
score=model.score(untouched_val_X,untouched_val_Y)
print(score)
predict = model.predict(untouched_test_X)

model= RandomForestClassifier(n_estimators=30, max_depth=4)
model.fit(untouched_train_X, untouched_train_Y)
score=model.score(untouched_val_X,untouched_val_Y)
print(score)
predict = model.predict(untouched_test_X)

model= RandomForestClassifier(n_estimators=30, max_depth=16)
model.fit(untouched_train_X, untouched_train_Y)
score=model.score(untouched_val_X,untouched_val_Y)
print(score)
predict = model.predict(untouched_test_X)

```