CS498HW1

Xinyi Gu xinyigu2

Three accuracies for 1A, 1B, 1D

1A:0.7477124

1B:0.7601307

1D:0.7352941

Screenshot of your code

```r
setwd('/Users/Sunny/Desktop/CS498/HW1')
raw_data<-read.csv('pima-indians-diabetes.csv', header=FALSE)
library(klaR)
library(caret)
bigx<-wdat[,-c(9)]
bigy<-wdat[,9]
trscore<-array(dim=10)
tescore<-array(dim=10)
for (wi in 1:10)
{wtd<-createDataPartition(y=bigy, p=.8, list=FALSE)
    nbx<-bigx
    ntrbx<-nbx[wtd, ]
    ntrby<-bigy[wtd]
    trposflag<-ntrby>0
    ptregs<-ntrbx[trposflag, ]
    ntregs<-ntrbx[!trposflag,]
    ntebx<-nbx[-wtd, ]
    nteby<-bigy[-wtd]
    ptrmean<-sapply(ptregs, mean, na.rm=TRUE)
    ntrmean<-sapply(ntregs, mean, na.rm=TRUE)
    ptrsd<-sapply(ptregs, sd, na.rm=TRUE)
    ntrsd<-sapply(ntregs, sd, na.rm=TRUE)
    ptroffsets<-t(t(ntrbx)-ptrmean)
    ptrscales<-t(t(ptroffsets)/ptrsd)
    ptrlogs<--(1/2)*rowSums(apply(ptrscales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ptrsd))
    ntroffsets<-t(t(ntrbx)-ntrmean)
    ntrscales<-t(t(ntroffsets)/ntrsd)
    ntrlogs<--(1/2)*rowSums(apply(ntrscales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ntrsd))
    lvwtr<-ptrlogs>ntrlogs
    gotrighttr<-lvwtr==ntrby
    trscore[wi]<-sum(gotrighttr)/(sum(gotrighttr)+sum(!gotrighttr))
    pteoffsets<-t(t(ntebx)-ptrmean)
    ptescales<-t(t(pteoffsets)/ptrsd)
    ptelogs<--(1/2)*rowSums(apply(ptescales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ptrsd))
    nteoffsets<-t(t(ntebx)-ntrmean)
    ntescales<-t(t(nteoffsets)/ntrsd)
    ntelogs<--(1/2)*rowSums(apply(ntescales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ntrsd))
    lvwte<-ptelogs>ntelogs
    gotright<-lvwte==nteby
    tescore[wi]<-sum(gotright)/(sum(gotright)+sum(!gotright))
}
    sum(tescore)/length(tescore)
```

```r
setwd('/Users/Sunny/Desktop/CS498/HW1')
wdat<-read.csv('pima-indians-diabetes.csv', header=FALSE)
library(klaR)
library(caret)
bigx<-wdat[,-c(9)]
bigy<-wdat[,9]
nbx<-bigx
for (i in c(3, 4, 6, 8))
{vw<-bigx[, i]==0
    nbx[vw, i]=NA
}
trscore<-array(dim=10)
tescore<-array(dim=10)
for (wi in 1:10)
{wtd<-createDataPartition(y=bigy, p=.8, list=FALSE)
    ntrbx<-nbx[wtd, ]
    ntrby<-bigy[wtd]
    trposflag<-ntrby>0
    ptregs<-ntrbx[trposflag, ]
    ntregs<-ntrbx[!trposflag,]
    ntebx<-nbx[-wtd, ]
    nteby<-bigy[-wtd]
    ptrmean<-sapply(ptregs, mean, na.rm=TRUE)
    ntrmean<-sapply(ntregs, mean, na.rm=TRUE)
    ptrsd<-sapply(ptregs, sd, na.rm=TRUE)
    ntrsd<-sapply(ntregs, sd, na.rm=TRUE)
    ptroffsets<-t(t(ntrbx)-ptrmean)
    ptrscales<-t(t(ptroffsets)/ptrsd)
    ptrlogs<--(1/2)*rowSums(apply(ptrscales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ptrsd))
    ntroffsets<-t(t(ntrbx)-ntrmean)
    ntrscales<-t(t(ntroffsets)/ntrsd)
    ntrlogs<--(1/2)*rowSums(apply(ntrscales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ntrsd))
    lvwtr<-ptrlogs>ntrlogs
    gotrighttr<-lvwtr==ntrby
    trscore[wi]<-sum(gotrighttr)/(sum(gotrighttr)+sum(!gotrighttr))
    pteoffsets<-t(t(ntebx)-ptrmean)
    ptescales<-t(t(pteoffsets)/ptrsd)
    ptelogs<--(1/2)*rowSums(apply(ptescales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ptrsd))
    nteoffsets<-t(t(ntebx)-ntrmean)
    ntescales<-t(t(nteoffsets)/ntrsd)
    ntelogs<--(1/2)*rowSums(apply(ntescales,c(1, 2), function(x)x^2), na.rm=TRUE)-sum(log(ntrsd))
    lvwte<-ptelogs>ntelogs
    gotright<-lvwte==nteby
    tescore[wi]<-sum(gotright)/(sum(gotright)+sum(!gotright))
}
```

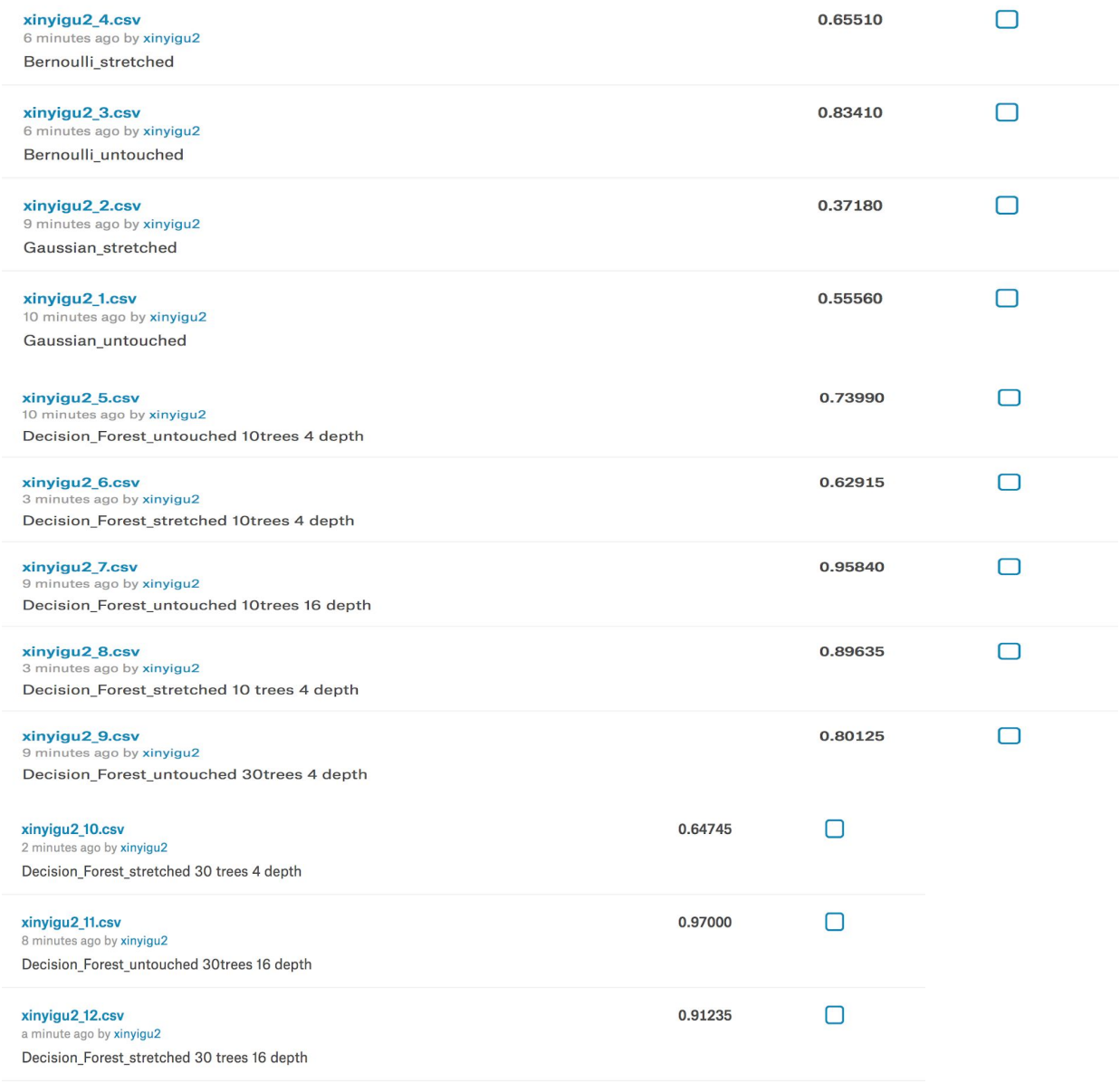
```r
setwd('/Users/Sunny/Desktop/CS498/HW1')
rm(list=ls())
wdat<-read.csv('pima-indians-diabetes.csv', header=FALSE)
library(klaR)
library(caret)
bigx<-wdat[,-c(9)]
bigx2<-apply(bigx, c(1, 2), function(x)x^2)
bigx<-cbind(bigx, bigx2)
errs<-array(dim=10)
cvs<-c(0.005, 0.01, 0.1)
for (wi in c(1:10))
{bigy<-as.factor(wdat[,9])
    wtd<-createDataPartition(y=bigy, p=.8, list=FALSE)
    wstring<-paste("-c", sprintf('%f', cvs[wi]), sep=" ")
    svm<-svmlight(bigx[wtd,], bigy[wtd], pathsvm='/Users/Sunny/Desktop/CS498/HW1/svm_light_OS10.8.4_i7/')
    labels<-predict(svm, bigx[-wtd,])
    foo<-labels$class
    errs[wi]<-sum(foo==bigy[-wtd])/(sum(foo==bigy[-wtd])+sum(!(foo==bigy[-wtd])))
}
mean(errs)
```

reference:http://luthuli.cs.uiuc.edu/~daf/courses/AML-18/RCodeClassification/svmholdoutsquaredfeats.Randhttp://luthuli.cs.uiuc.edu/~daf/courses/AML-18/RCodeClassification/pimanbholdout.R

Table of accuracies for all 12 cases.

Method	Gaussian	Bernoulli	DF_10_4	DF_10_16	DF_30_4	DF_30_16
Untouched	0.5535	0.833	0.7545	0.968	0.793	0.9775
Stretched	0.3705	0.6465	0.6205	0.902	0.651	0.917

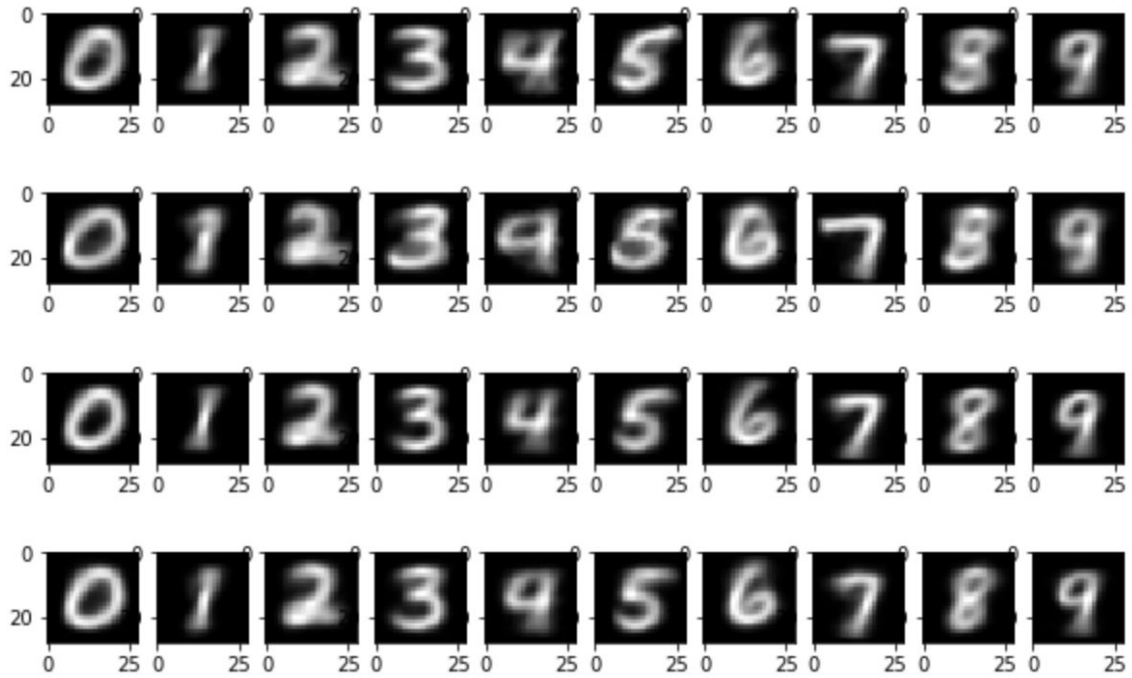
Screenshot of Kaggle (READABLE SCREENSHOT)



A brief explanation of which model is better and why

I think decision forest classifier is better if proper number of trees and depth are chosen. It is easy to interpret and we can apply different numbers to see under what trees and depth this model perform the best. The untouched data set works,in most cases, better than the stretched data set and this is probably because the boundary condition.

40 mean images (4x10 of part 2A)



Screenshot of your code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from collections import Counter
import csv
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import BernoulliNB
from sklearn.ensemble import RandomForestClassifier
def bounding_box(u):
    minx = 27
    miny = 27
    maxx = 0
    maxy = 0
    u = np.reshape(u, (28, 28))
    for x in range (28):
        for y in range (28):
            if u[x][y] > 0:
                if x < minx:
                    minx = x
                if y < miny:
                    miny = y
                if x > maxx:
                    maxx = x
                if y > maxy:
                    maxy = y
    v = u[minx:maxx+1][miny:maxy+1]
    v = np.resize(v,(20,20))
    v = np.reshape(v, (400))
    return v
def array_to_csv(name,array):
    with open(name,'wt') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(('ImageId','Label'))
        for i in range(len(array)):
            writer.writerow((i , array[i]))
def Gaussian_NB(train_set,train_label,test_set,val_set,val_label):
    classifier = GaussianNB()
    classifier.fit(train_set,train_label)
    accuracy = accuracy_score(val_label, classifier.predict(val_set))
    print(accuracy)
    return classifier.predict(test_set)
def Bernoulli_NB(train_set,train_label,test_set,val_set,val_label):
    classifier = BernoulliNB()
    classifier.fit(train_set,train_label)
```

```
def Bernoulli_NB(train_set,train_label,test_set,val_set,val_label):
    classifier = BernoulliNB()
    classifier.fit(train_set,train_label)
    accuracy = accuracy_score(val_label, classifier.predict(val_set))
    print(accuracy)
    return classifier.predict(test_set)
def Decision_Forest(train_set,train_label,test_set,trees,depth,val_set,val_label):
    classifier = RandomForestClassifier(n_estimators =trees, max_depth=depth)
    classifier.fit(train_set,train_label)
    accuracy = accuracy_score(val_label, classifier.predict(val_set))
    print(accuracy)
    return classifier.predict(test_set)
def drawpic(test_label, test_data):
    num = []
    array = np.zeros((10, 28*28))
    count = Counter(test_label)
    for i in range(10):
        num.append(count[i])

    for i in range(20000):
        label = test_label[i]
        for j in range(28*28):
            if(test_data[i][j] > 0):
                array[label][j] += int(test_data[i][j])/(num[label]*255)
    draw_figure =plt.figure(figsize=(10,10))
    for i in range(10):
        draw_figure.add_subplot(1, 10, i+1)
        plt.imshow(array[i].reshape((28,28)), cmap="gray")
    plt.show()
train_set = pd.read_csv("/Users/Sunny/Desktop/CS498/HW1/train.csv").as_matrix()
test_set = pd.read_csv("/Users/Sunny/Desktop/CS498/HW1/test.csv",header=None).as_matrix()
val_set = pd.read_csv("/Users/Sunny/Desktop/CS498/HW1/val.csv").as_matrix()
train_label = train_set[:,1]
train_untouched =train_set[:,2:]
train_stretched = [bounding_box(row) for row in train_untouched]
val_label = val_set[:,0]
val_untouched = val_set[:,1:]
val_stretched = [bounding_box(row.reshape(784)) for row in val_untouched]
Gaussian_untouched  = Gaussian_NB(train_untouched,train_label,test_set,val_untouched,val_label)
Gaussian_stretched  = Gaussian_NB(train_stretched,train_label,test_stretched,val_stretched,val_label)
Bernoulli_untouched = Bernoulli_NB(train_untouched,train_label,test_set,val_untouched,val_label)
Bernoulli_stretched = Bernoulli_NB(train_stretched,train_label,test_stretched,val_stretched,val_label)
Decision_Forest_untouched104 = Decision_Forest(train_untouched,train_label,test_set,
    10,4,val_untouched,val_label)
Decision_Forest_stretched104 = Decision_Forest(train_stretched,train_label,test_stretched,
    10,4,val_stretched,val_label)
```

```
Gaussian_untouched  = Gaussian_NB(train_untouched,train_label,test_set,val_untouched,val_label)
Gaussian_stretched  = Gaussian_NB(train_stretched,train_label,test_stretched,val_stretched,val_label)
Bernoulli_untouched = Bernoulli_NB(train_untouched,train_label,test_set,val_untouched,val_label)
Bernoulli_stretched = Bernoulli_NB(train_stretched,train_label,test_stretched,val_stretched,val_label)
Decision_Forest_untouched104 = Decision_Forest(train_untouched,train_label,test_set,
    10,4,val_untouched,val_label)
Decision_Forest_stretched104 = Decision_Forest(train_stretched,train_label,test_stretched,
    10,4,val_stretched,val_label)
Decision_Forest_untouched1016 = Decision_Forest(train_untouched,train_label,test_set,
    10,16,val_untouched,val_label)
Decision_Forest_stretched1016 = Decision_Forest(train_stretched,train_label,test_stretched,
    10,16,val_stretched,val_label)
Decision_Forest_untouched304 = Decision_Forest(train_untouched,train_label,test_set,
    30,4,val_untouched,val_label)
Decision_Forest_stretched304 = Decision_Forest(train_stretched,train_label,test_stretched,
    30,4,val_stretched,val_label)
Decision_Forest_untouched3016 = Decision_Forest(train_untouched,train_label,test_set,
    30,16,val_untouched,val_label)
Decision_Forest_stretched3016 = Decision_Forest(train_stretched,train_label,test_stretched,
    30,16,val_stretched,val_label)

drawpic(Gaussian_untouched, test_set)
drawpic(Gaussian_stretched, test_set)
drawpic(Bernoulli_untouched, test_set)
drawpic(Bernoulli_stretched, test_set)
array_to_csv('xinyigu2_1.csv',Gaussian_untouched)
array_to_csv('xinyigu2_2.csv',Gaussian_stretched)
array_to_csv('xinyigu2_3.csv',Bernoulli_untouched)
array_to_csv('xinyigu2_4.csv',Bernoulli_stretched)

array_to_csv('xinyigu2_5.csv',Decision_Forest_untouched104)
array_to_csv('xinyigu2_6.csv',Decision_Forest_stretched104)
array_to_csv('xinyigu2_7.csv',Decision_Forest_untouched1016)
array_to_csv('xinyigu2_8.csv',Decision_Forest_stretched1016)
array_to_csv('xinyigu2_9.csv',Decision_Forest_untouched304)
array_to_csv('xinyigu2_10.csv',Decision_Forest_stretched304)
array_to_csv('xinyigu2_11.csv',Decision_Forest_untouched3016)
array_to_csv('xinyigu2_12.csv',Decision_Forest_stretched3016)
```