FINAL PROJECT: FINAL REPORT

Fadhil Salih

DECEMBER 11, 2018

STAT-517

UNIVERSITY OF IDAHO

ABSTRACT

I have chosen the HAM10000 dataset for my final project. It is abbreviated as humans against machines with 10000 dermatoscopic images. Each record is a dermatoscopic image of pigmented facial lesions collected from different populations. Each image, represented by 64 pixels (8X8), is classified into different labels. Each label represents an important diagnostic category in the realm of pigmented lesions. This dataset will be used as a training dataset for performing academic machine learning practices. Supervised (Classification) and unsupervised (Clustering and associations) learning practices will be employed on this dataset to produce useful information.

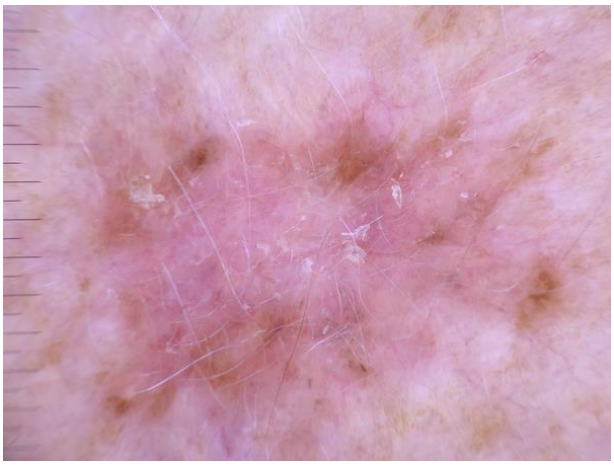
INTRODUCTION

Dermatoscopy is a widely used diagnostic technique that improves the diagnosis of benign and malignant pigmented skin lesions in comparison to examination with the unaided eye. Dermatoscopic images are also a suitable source to train artificial neural networks to diagnose pigmented skin lesions automatically. Training of neural networks for automated diagnosis of pigmented skin lesions used to be hampered by the small size and lack of diversity of available dataset of dermatoscopic images.

The HAM10000 dataset consists of 10015 dermatoscopic images which can serve as a training set for academic machine learning purposes. Tschandl, Rosendahl, and Kittler (2015) researched on this dataset to better characterize dermatoscopic patterns of several types of flat pigmented facial lesions and to analyze their significance by calculating their relative risks and diagnostic values. They prospectively analyzed dermatoscopic images of 240 flat pigmented facial skin lesions collected consecutively from 195 patients (41.5% females, mean age:

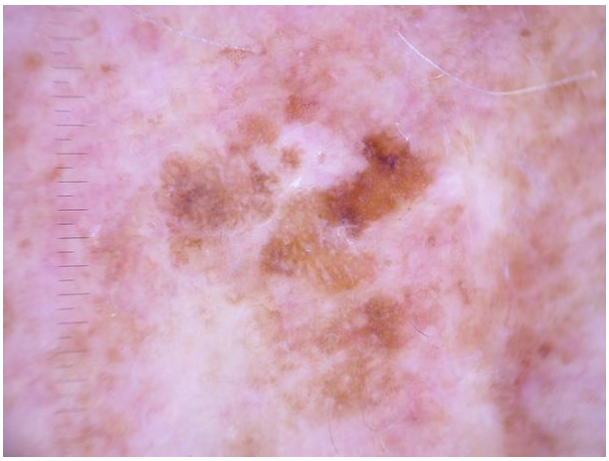
61 ± 14 years) between 2007 and March 2012 in a primary skin cancer practice situated in Queensland, Australia. Cases include a representative collection of all important diagnostic categories in the realm of pigmented lesions: Actinic keratoses and intraepithelial carcinoma / Bowen's disease (**akiec**), basal cell carcinoma (**bcc**), benign keratosis-like lesions (solar lentigines / seborrheic keratoses and lichen-planus like keratoses, **bkl**), dermatofibroma (**df**), melanocytic nevi (**nv**), vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, **vasc**), and melanoma (**mel**).The diagnosis of flat pigmented lesions on the face is challenging because of the morphologic overlap of biologically different lesions and the unknown significance of dermatoscopic patterns. More than 50% of lesions are confirmed through histopathology (histo), the ground truth for the rest of the cases is either follow-up examination (follow_up), expert consensus (consensus), or confirmation by in-vivo confocal microscopy (confocal).

Diagnostic categories for facial lesions



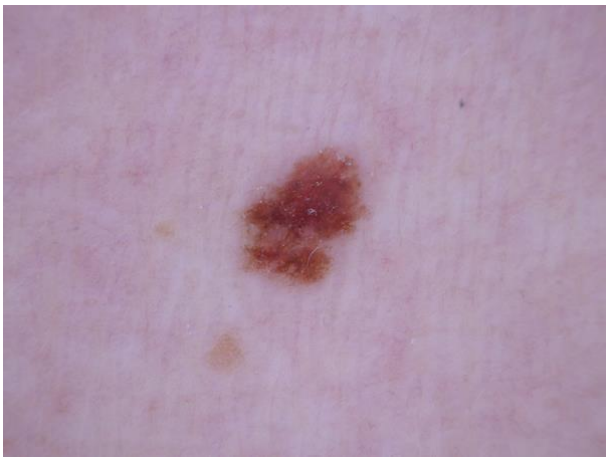
Label:0, akiec

Label:1, bcc



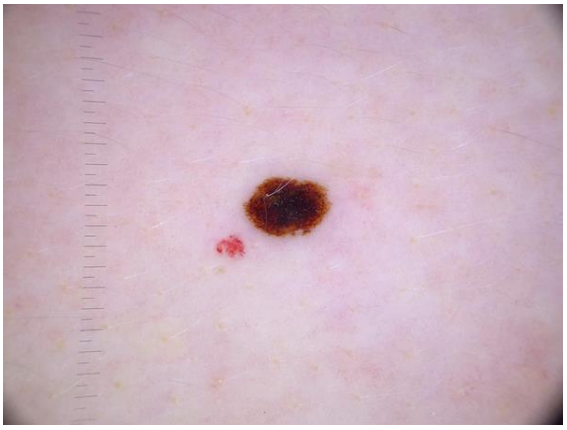
Label:2, bkl

Label:3, df



Label:4, nv

Label:5, vasc



Label:6, mel

METHODOLOGY

HAM10000 Dataset:

10015 X 193 – 10015 images and 192 vectorized pixel (RGB 8 X 8) values between 0 and 255, along with corresponding labels of diagnostic information.

	A	B	C	D	E	F	G	H	I
1	pixel0000	pixel0001	pixel0002	pixel0003	pixel0004	pixel0005	pixel0006	pixel0007	pixel0008
2	199	156	188	210	165	198	216	176	203
3	115	87	115	180	133	158	200	153	187
4	203	146	164	209	144	153	217	160	181
5	135	96	110	198	141	156	208	148	158
6	204	156	187	232	185	216	236	196	220
7	26	13	18	175	147	152	247	220	231
8	201	156	149	204	157	150	206	155	151
9	174	129	118	189	139	136	201	147	140
10	175	122	127	199	138	152	214	159	181
11	220	163	175	216	152	169	230	172	187
12	53	43	45	153	134	138	179	160	170
13	172	142	143	181	147	152	183	147	153

For this project, I have used the HAM10000 dataset to perform useful machine learning practices. Each record represents a dermatoscopic image of pigmented facial lesions, containing pixel information (8X8). The target variable in this case is the label. The label represents the diagnostic category of the pigmented facial lesion. The categories are actinic keratoses and intraepithelial carcinoma / Bowen's disease, basal cell carcinoma (bcc), benign keratosis-like lesions (solar lentigines / seborrheic keratoses and lichen-planus like keratoses, bkl), dermatofibroma (df), melanocytic nevi (nv), melanoma (mel), and vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, vasc). They are labelled from 0 to 6 respectively in the HAM10000 dataset.

10015 dermatoscopic images were used for the study, out of which 7511 images were used to train and 2504 images were used to test the model using python (train test split-75:25). This was used for classification under supervised learning methods such as Gaussian naïve bayes,

support vector machines, neural network (Multi-layer perceptron), random forest, decision tree, and k-nearest neighbor. Clustering analysis was performed on the dataset to demonstrate an example of unsupervised learning using k-means, fuzzy k-means, hierarchical clustering, nbclust and mclust. Association rules were demonstrated on the dataset using apriori algorithm to produce useful information. Both clustering and association were performed in R.

RESULTS

Preliminary results indicated that label 4, i.e. melanocytic nevi (**nv)**, is the diagnostic category that most of the images were placed under followed by label 6, i.e. melanoma (**mel**).

```
4    6705
6    1113
2    1099
1     514
0     327
5     142
3     115
Name: label, dtype: int64
```

Classification

Aim: To classify each record or image to its corresponding label from the vectorized pixel dataset using different classifiers

Code:

```
y = pd.DataFrame([skin.label_new]).T # Separating the response variable
x = pd.DataFrame(skin)
x = skin.drop(['label_new'], axis = 1)
from sklearn.cross_validation import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, random_state = 7, test_size = 0.25)
#Gaussian Naïve Bayes
from sklearn.naive_bayes import GaussianNB
```

```
GNB_model = GaussianNB()
GNB_model.fit(xtrain, ytrain)
y_mod = GNB_model.predict(xtest)
accuracy_GNB1 = round(GNB_model.score(xtrain, ytrain) * 100, 2)
accuracy_GNB2 = round(accuracy_score(ytest, y_mod) * 100, 2)
#Support vector machines
SVC_model = SVC() # Calling the support vector machine function
SVC_model.fit(xtrain, ytrain)
y_mod = SVC_model.predict(xtest)
accuracy_SVC1 = round(SVC_model.score(xtrain, ytrain) * 100, 2)
accuracy_SVC2 = round(accuracy_score(ytest, y_mod) * 100, 2)
# Binarize the output
y = label_binarize(y, classes=[0, 1, 2, 3, 4, 5, 6])
n_classes = y.shape[1]
#Splitting the binarized dataset
from sklearn.cross_validation import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, random_state = 7, test_size = 0.25)
#K-nearest neighbor
Knn_Model = KNeighborsClassifier(n_neighbors = 3)
Knn_Model.fit(xtrain, ytrain) #Fitting the model
y_mod = Knn_Model.predict(xtest) #Predicting response variable
accuracy_Knn1 = round(Knn_Model.score(xtrain, ytrain) * 100, 2) # Testing the accuracy
between xtrain and ytrain
accuracy_Knn2 = round(accuracy_score(ytest, y_mod) * 100, 2) # Testing the accuracy between
ytest and y_pred
auc1= roc_auc_score(ytest, y_mod) # Generates area under ROC curve
#Decision Tree
Dtree_model = DecisionTreeClassifier()
Dtree_model.fit(xtrain, ytrain)
y_mod = Dtree_model.predict(xtest)
```

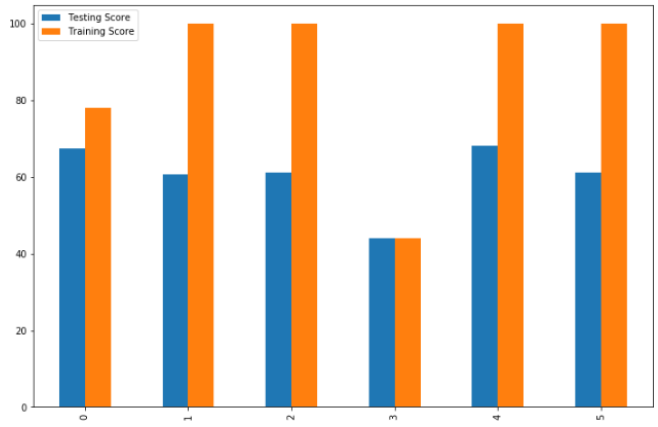
```
accuracy_Dtree1 = round(Dtree_model.score(xtrain, ytrain) * 100, 2)
accuracy_Dtree2 = round(accuracy_score(ytest, y_mod) * 100, 2)
auc2= roc_auc_score(ytest, y_mod) # Generates area under ROC curve
#Random Forest
skin_forest = RandomForestClassifier(n_estimators = 100, random_state = 0)
skin_forest.fit(xtrain, ytrain)
y_mod = skin_forest.predict(xtest)
accuracy_random_forest1 = round(skin_forest.score(xtrain, ytrain) * 100, 2)
accuracy_random_forest2 = round(accuracy_score(ytest, y_mod) * 100, 2)
auc3= roc_auc_score(ytest, y_mod) # Generates area under ROC curve
#Neural network: Multi-Layer Perceptron (MLP)
MLP_model = MLPClassifier(random_state=57)
MLP_model.fit(xtrain, ytrain)
y_mod = MLP_model.predict(xtest)
accuracy_MLP1 = round(skin_forest.score(xtrain, ytrain) * 100, 2)
accuracy_MLP2 = round(skin_forest.score(xtest, ytest) * 100, 2)
auc4= roc_auc_score(ytest, y_mod) # Generates area under ROC curve
#Displaying results
models_summary1 = pd.DataFrame({
   'Model': ['KNN', 'Decision Tree','Random Forest', 'Gaussian naive Bayes', 'Support vector machines', 'MLP Classifier'],
   'Training Score': [accuracy_Knn1, accuracy_Dtree1, accuracy_random_forest1, accuracy_GNB1, accuracy_SVC1, accuracy_MLP1],
   'Testing Score': [accuracy_Knn2, accuracy_Dtree2, accuracy_random_forest2, accuracy_GNB2, accuracy_SVC2, accuracy_MLP2],
    })
models_summary1
models_summary1.plot(kind = 'bar', figsize = (12, 8))
models_summary2 = pd.DataFrame({
   'Model': ['KNN', 'Decision Tree','Random Forest', 'MLP Classifier'],
   'ROC Score': [auc1, auc2, auc3, auc4],
```

```
    })
models_summary2
```

Output:

	Model	Testing Score	Training Score
0	KNN	67.45	78.02
1	Decision Tree	60.62	100.00
2	Random Forest	61.30	99.99
3	Gaussian naive Bayes	44.05	43.98
4	Support vector machines	68.13	100.00
5	MLP Classifier	61.30	99.99

	Model	ROC Score
0	KNN	0.586285
1	Decision Tree	0.588462
2	Random Forest	0.549616
3	MLP Classifier	0.541899



Classification results indicated that support vector machines had the best accuracy score for both training and testing datasets. Decision tree had the highest ROC score among other classifiers. Support vector machines and Gaussian naïve bayes classifier could not accept multi-label/class datasets. Even after binarizing the dataset, these two classifiers could not accept the dataset due to bad input shape error. Therefore, the ROC score for support vector machines and Gaussian naïve bayes classifier was not obtained.

Clustering

Aim: To cluster the dataset according to its 7 labels and derive the optimal number of clusters to validate with required number of clusters

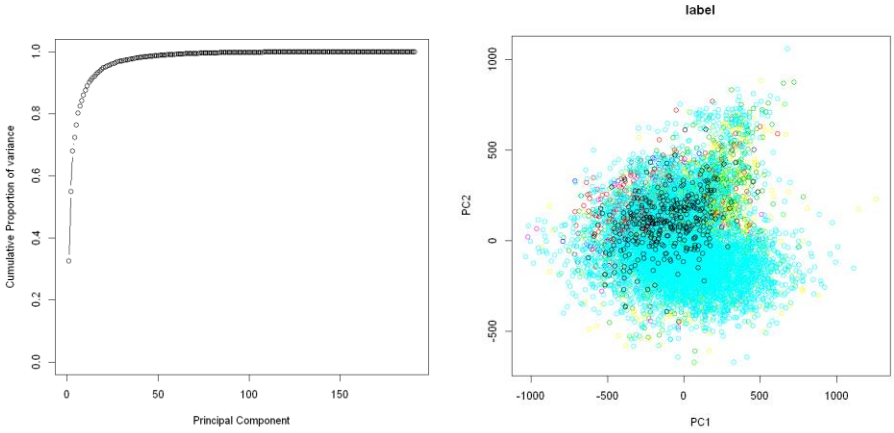
Principal component analysis (PCA)

Code:

```
sk <- skin[1:192] # Removing label information
par(mfrow=c(1,2))
skin.s=scale(sk)
skin.p=prcomp(sk[,-1],scale=FALSE)
# Loading the principal components
skin.p$rotation[1:10,1:10]
# Generating the rotation matrix for analysis
sk.sd=skin.p$sdev
sk.var=skin.p$sdev^2
sk.var[1:5]
# Computing standard deviation and variance for each principal component
pvar=sk.var/sum(sk.var)
which.max(cumsum(pvar)[cumsum(pvar)<0.95])
which.max(cumsum(pvar)[cumsum(pvar)<0.98])
which.max(cumsum(pvar)[cumsum(pvar)<0.80])
# Calculating the number of PC's required to retain 80, 95 and 98 percent of the total variance
# First 20 principal components
cumsum(pvar[20])
plot(cumsum(pvar),xlab="Principal Component",
  ylab="Cumulative Proportion of variance",ylim=c(0,1),type='b')
# Arrow head length is suppressed to get rid of the errors of indeterminate angle
skingroups1<- factor(skin$label)
plot(main="label",skin.p$x[,1:100],col=skingroups1)
skin_new1=skin.p$x[,1:100]
skin_new1.s=scale(skin_new1)
```

Output:

```
73057.22158748  50176.69835542  29273.3655746594  9745.36844911681  9008.87792374357

20

38

5
```



PCA indicated that 20 PC's retains 95% of the total variance and 38 PC's retain 98% of the total variance. 20 PC's were selected for clustering analysis.

1. K-means

   Code:

```
results <- kmeans(skin.s, 7)
results
```

   Output:

```
K-means clustering with 7 clusters of sizes 481, 1619, 1034, 2416, 1637, 1664, 1164

Cluster means:
    pixel0000    pixel0001   pixel0002   pixel0003   pixel0004   pixel0005
1 -2.62879182 -2.503610620 -2.46134952 -2.28265327 -1.82543682 -1.79165063
2 -0.85230843 -0.507335671 -0.48442522 -1.07557146 -0.45568673 -0.41876714
3  0.08128691  0.696532591  0.72858070  0.02156445  0.80968169  0.81717983
4  0.79218586  0.246166029  0.24498708  0.89040596  0.04518619  0.06085034
5  0.46369056 -0.319039271 -0.31852449  0.36743658 -0.80765354 -0.80049781
6 -0.42518998 -0.008158832 -0.03538979 -0.41115576  0.26069724  0.22498668
7  0.51101304  1.070880512  1.03373327  0.64300470  1.33826449  1.27476518

Clustering vector:
   [1] 6 6 6 6 6 7 1 6 6 6 7 6 6 6 6 6 6 7 6 6 6 3 1 6 3 6 2 3 2 6 3 3 1 2 6 7 1 2
  [37] 2 7 6 2 7 3 6 6 2 2 2 6 7 6 2 2 2 1 2 6 2 2 1 6 1 2 1 2 6 2 2 2 2 2 2 1
  [73] 2 2 1 6 6 3 1 2 1 2 1 2 3 1 3 2 2 6 1 1 2 2 2 2 2 2 1 1 2 2 1 2 6 1 2 2
 [109] 2 2 2 2 1 2 1 7 7 7 7 2 2 3 3 2 2 3 2 1 1 2 3 6 2 3 2 2 1 2 2 6 3 2 2 6
 [145] 2 2 3 3 3 1 1 3 2 1 6 3 1 2 1 2 1 2 1 2 1 2 3 3 3 1 3 2 2 1 6 6 7 3 1 3
 [181] 2 2 3 2 1 1 3 1 2 6 1 1 2 3 2 2 6 3 1 3 2 2 2 1 2 3 3 3 2 6 2 3 3 7 7 7
 [217] 3 3 2 2 1 1 6 3 3 1 6 1 3 6 2 2 1 2 3 2 5 1 2 2 6 2 3 6 6 2 2 2 2 3 2 2

Within cluster sum of squares by cluster:
[1]  67418.19 133876.82 158559.15 176627.60 164262.28 115008.26 127811.51
 (between_SS / total_SS =  50.9 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

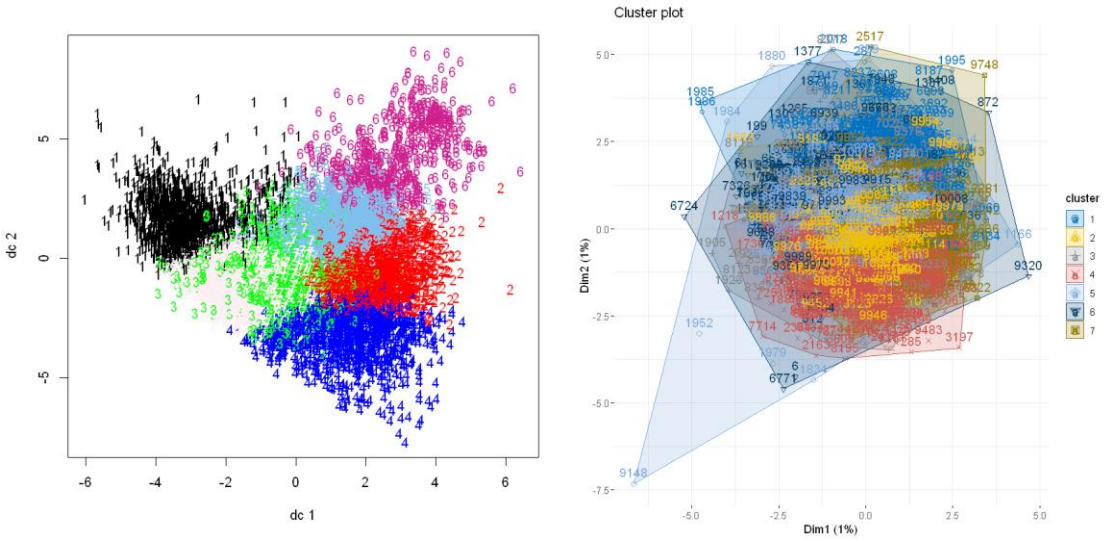
K-means clustering was done on 20 PC's generating 7 clusters (labels) as required. The sum of squares value was 50.9%, which is relatively low. This indicates that k-means is not a good fit for this dataset.

2. Fuzzy K-means

Code:

```
set.seed(10)
kmeans_7.fit=kmeans(skin_new1,7,nstart=50)
attributes(kmeans_7.fit)
kmeans_7.fit$size
kmeans_7.fit$tot.withinss
plotcluster(skin_new1,kmeans_7.fit$cluster)
fviz_cluster(kmeans_7.fit,data=skin_new1,ellipse.type="convex",palette="jco",ggtheme=
theme_minimal())
```

Output:



$names
'cluster' 'centers' 'totss' 'withinss' 'tot.withinss' 'betweenss' 'size' 'iter' 'ifault'

$class
'kmeans'

1583 1540 1133 1153 1564 567 2475

1148122974.52263

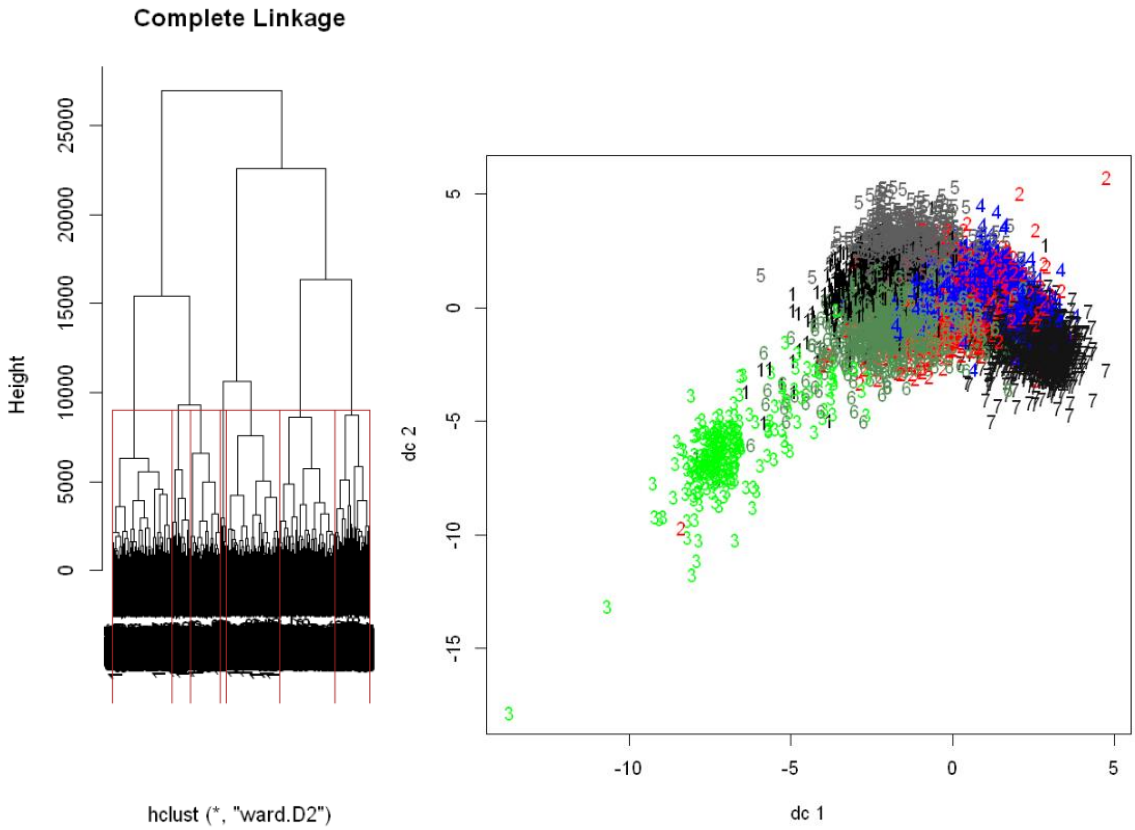
Fuzzy k-means clustering was done on 20 PC's, generating 7 clusters (labels) as required. One cluster seems to be missing in the results shown above. It is assumed to be superimposed by other colors as 7 clusters have clearly been generated. The objective function value was seen to be very high, which is undesirable, as we need to minimize intra-cluster variance.

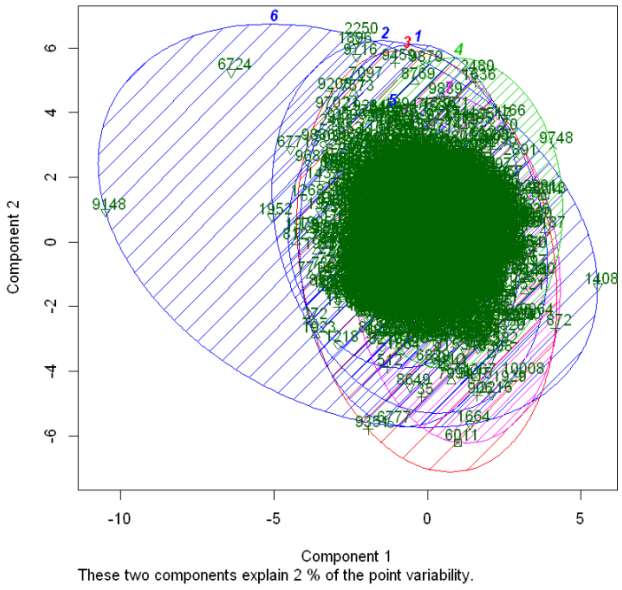
3. Hierarchical clustering:

Code:

```
par(mfrow=c(1,2))
#Hierarchial Clustering with K=7
skhc_ward=hclust(dist(skin_new1,method="euclidean"),method="ward.D2")
plot(skhc_ward, main="Complete Linkage",xlab="",cex=.9) #Dendogram
rect.hclust(skhc_ward,k=7,border="Brown") #Bordering the clusters
```

Output:





Results from hierarchical clustering indicated that the chosen number of clusters (K = 7) are acceptable for analysis

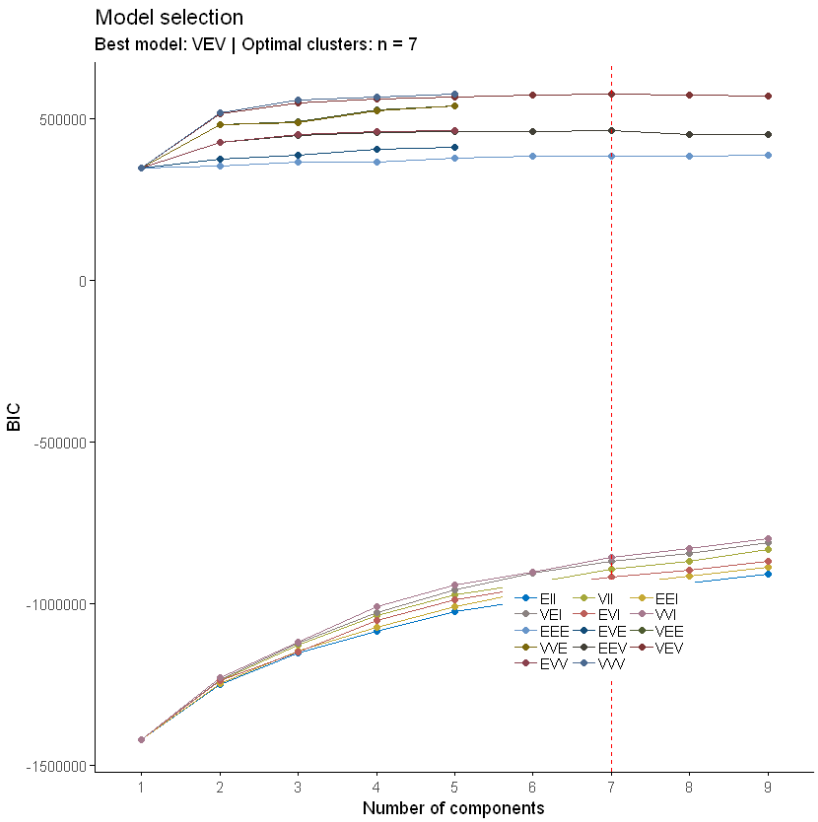
4. Model based clustering

Code:

```
# Generating the optimal number of clusters
par(mfrow=c(1,3))
sk_fit<-Mclust(skin.s[,0:50])
summary(sk_fit)
sk_fit$modelName
sk_fit$G
fviz_mclust(sk_fit,"BIC",palette="jco")
```

Output:

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VEV (ellipsoidal, equal shape) model with 7 components:

 log.likelihood     n   df      BIC      ICL
       328301.2 10015 8987 573815.6 573339.6

Clustering table:
   1    2    3    4    5    6    7
2032 1995 1695 1965  953 1230  145

'VEV'

7
```



MbClust results indicate that 7 is the optimal number of clusters using the VEV model, which is exactly the number of required clusters. This indicates that the model is appropriate.

5. NbClust:

   Code:

```
# To generate the number of optimal clusters
# Methods used (PC 1 - PC 20): NbClust, WSS, Gap statistic, and Silhouette
set.seed(10)
sk.nbclust<-skin.s %>%
  scale() %>%
  NbClust(distance="euclidean",min.nc=2,max.nc=8,method="complete",index="all")
fviz_nbclust(skin.s,kmeans,method="wss")
fviz_nbclust(skin.s,kmeans,method="gap_stat")
fviz_nbclust(skin.s,kmeans,method="silhouette")
```

Output:

```
*******************************************************************
* Among all indices:
* 5 proposed 2 as the best number of clusters
* 3 proposed 3 as the best number of clusters
* 6 proposed 4 as the best number of clusters
* 2 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters
* 2 proposed 8 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  4
```

Results indicated that 4 is the best number of clusters, according to majority rule. WSS, gap statistic and silhouette methods were executed to find the optimal number of clusters. Since the dataset is large, the code executed half way, without showing the results for WSS, gap statistic and silhouette method.

Associations

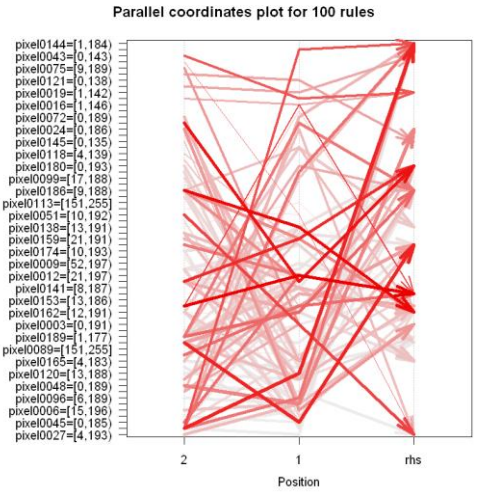
Aim: To generate associations for the entire dataset and between vectorized pixel values in different images to produce meaningful results.

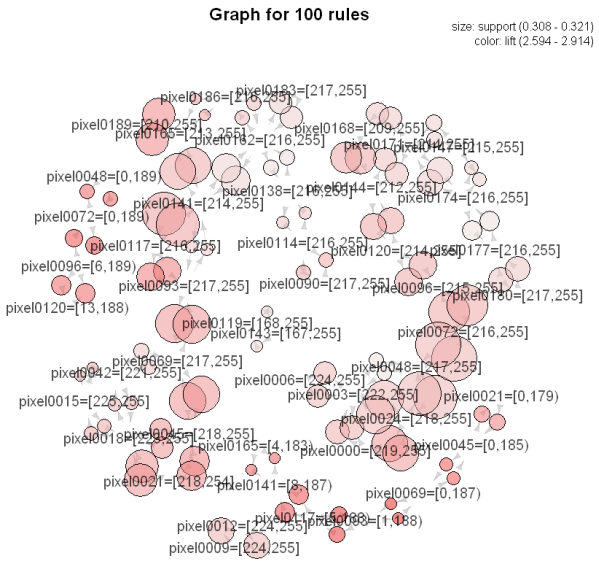
Code:

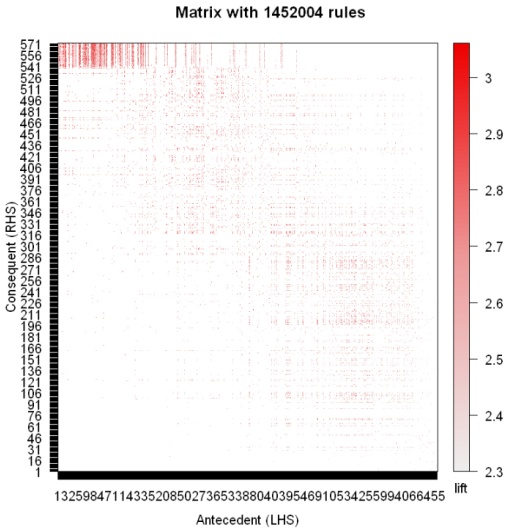
```
#Association done on the entire dataset
sk_dsz<-discretizeDF(sk)
sk_rules<-apriori(sk_dsz)
summary(sk_rules)
sk_subr1<-sk_rules[quality(sk_rules)$confidence>0.5]
plot(sk_subr1,method="matrix",measure = "lift")
sk_subr2<-head(sort(sk_rules,by="lift"), 100)
plot(sk_subr2, method="paracoord")
plot(sk_subr2,method = "graph")
```

Output:

```
set of 1452004 rules

rule length distribution (lhs + rhs):sizes
      2       3
   3232 1448772

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   3.000   3.000   2.998   3.000   3.000

summary of quality measures:
    support         confidence          lift            count
 Min.   :0.1000   Min.   :0.8000   Min.   :2.296   Min.   :1002
 1st Qu.:0.1105   1st Qu.:0.8282   1st Qu.:2.472   1st Qu.:1107
 Median :0.1256   Median :0.8583   Median :2.565   Median :1258
 Mean   :0.1429   Mean   :0.8628   Mean   :2.579   Mean   :1432
 3rd Qu.:0.1699   3rd Qu.:0.8923   3rd Qu.:2.675   3rd Qu.:1702
 Max.   :0.3206   Max.   :0.9980   Max.   :3.056   Max.   :3211

mining info:
    data ntransactions support confidence
 ASV_dsz         10015     0.1        0.8
```







Association done on the entire dataset generated a set of 1452004 rules. It essentially indicates the relationship between vectorized pixel intensities.

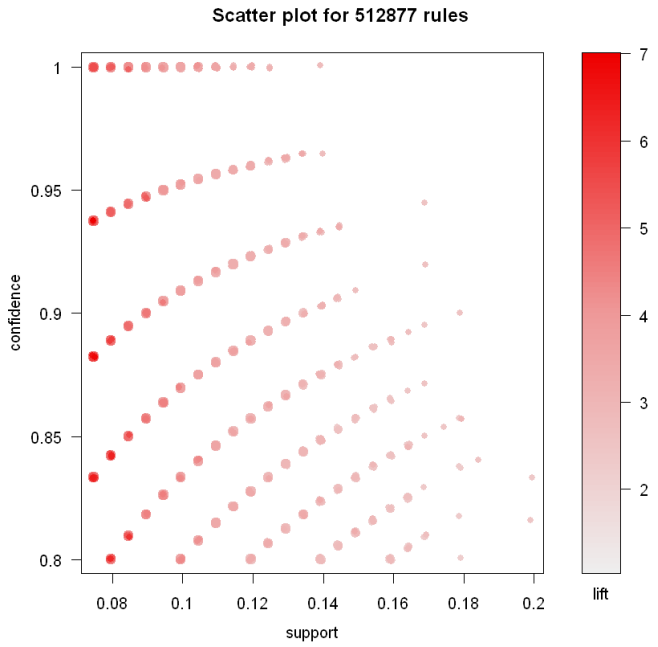
Code:

```
#Association done on two random pixels
skdata <- sk[,c(4,82)]
skdata$pixel0003 <- as.numeric(skdata$pixel0003)
skdata$pixel0001 <- as.numeric(skdata$pixel0081)
require("plyr")
sklist <- ddply(skdata,c("pixel0003"),
```

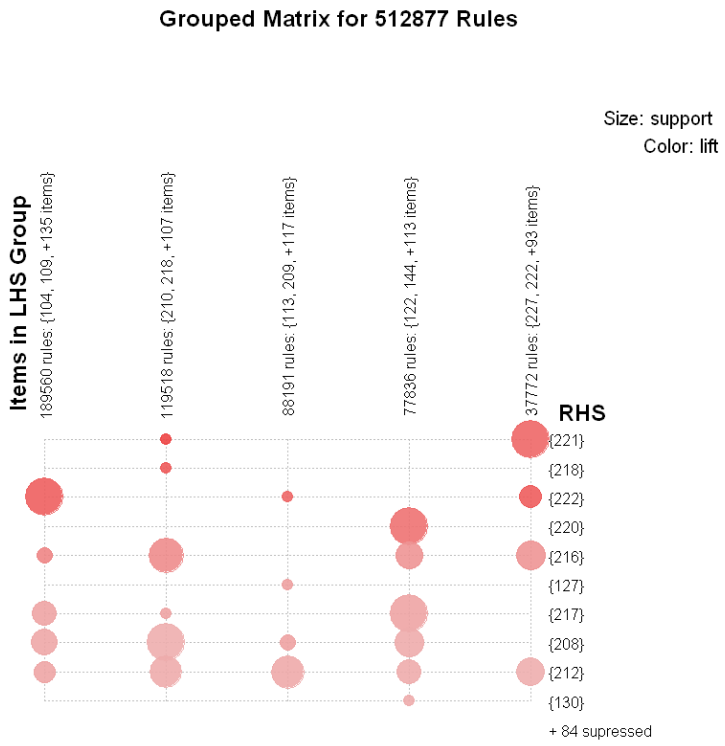
```
            function(df1)paste(df1$pixel0081,
                               collapse = ","))
sklist$pixel0003 <- NULL
write.csv(sklist,"C:/Users/fadhi/Desktop/Stat 517/Final Project/sklist.csv", quote=FALSE,
row.names=TRUE)
txn = read.transactions(file="C:/Users/fadhi/Desktop/Stat 517/Final Project/sklist.csv",
rm.duplicates=TRUE, format="basket", sep=",", cols=1)
txn@itemInfo$labels <- gsub("\"","",txn@itemInfo$labels)
skrules <- apriori(txn, parameter= list(sup = 0.07, conf=0.8, target = "rules"))
inspect(skrules)
plot(skrules)
plot(skrules, method="grouped", control = list(k=5))
```

Output:

```
     lhs                    rhs    support    confidence lift
[1]  {82}                => {156} 0.07462687 0.8333333  2.427536
[2]  {82}                => {168} 0.07462687 0.8333333  2.233333
[3]  {75}                => {171} 0.07462687 0.8333333  2.326389
[4]  {99}                => {145} 0.07462687 0.8333333  2.701613
[5]  {99}                => {177} 0.08457711 0.9444444  2.751208
[6]  {93}                => {157} 0.07960199 0.8421053  2.227147
[7]  {87}                => {168} 0.08457711 0.8095238  2.169524
[8]  {97}                => {171} 0.09452736 0.8636364  2.410985
[9]  {98}                => {192} 0.08457711 0.8500000  2.847500
[10] {98}                => {182} 0.07960199 0.8000000  2.593548
[11] {98}                => {190} 0.07960199 0.8000000  2.436364
[12] {98}                => {176} 0.07960199 0.8000000  2.172973
[13] {98}                => {187} 0.07960199 0.8000000  2.330435
[14] {98}                => {177} 0.08457711 0.8500000  2.476087
[15] {231}               => {195} 0.08457711 0.8095238  2.805419
[16] {230}               => {187} 0.09452736 0.8260870  2.406427
[17] {227}               => {202} 0.09452736 0.8260870  3.388642
[18] {227}               => {203} 0.09452736 0.8260870  3.388642
```



```
transactions in sparse format with
 201 transactions (rows) and
 246 items (columns)

[10480]  {200,203}                  => {183} 0.11442786 0.8214286  2.751786
[10481]  {200,203}                  => {193} 0.12935323 0.9285714  3.010369
[10482]  {200,203}                  => {157} 0.11442786 0.8214286  2.172462
[10483]  {200,203}                  => {177} 0.11442786 0.8214286  2.392857
[10484]  {200,203}                  => {171} 0.11940299 0.8571429  2.392857
[10485]  {161,200}                  => {183} 0.12935323 0.8125000  2.721875
[10486]  {161,200}                  => {193} 0.12935323 0.8125000  2.634073
[10487]  {158,200}                  => {183} 0.11940299 0.8571429  2.871429
```



Association was done between pixel intensities in pixel0003 and pixel0081 (Random 2 pixels). It shows the relationship between pixel intensities for certain images. For example, certain images have a similar set of pixel intensities that are followed by a particular pixel intensity.

CONCLUSION

Classification results indicated that support vector machines classifier had the best accuracy for this dataset. In terms of ROC curve, decision tree has the highest area under the roc curve. Clustering analysis revealed that the created clusters are somewhat acceptable for most models. Mbclust revealed that 7 is the optimal number of clusters which is the exact same number of required clusters. Association reveals the relationship between different vectorized pixels in the dataset with significant support, confidence and lift.

FUTURE WORK

Computers with better processors can run this dataset faster and generate more models to find the best fit. More research on the multiclass error may result in obtaining ROC scores for support vector machines and gaussian naïve bayes classifiers.

REFERENCES

Tschandl, P., Rosendahl, C., & Kittler, H. (2015). The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data.* 5. http://dx.doi.org/10.1038/sdata.2018.161

https://www.kaggle.com/kmader/skin-cancer-mnist-ham10000/home