STAT 6500

Statistical Machine Learning

Land Use Cover - EDA

Kendall Byrd - Atitarn Dechasuravanit - Alexys Rodriguez - Abdulaziz Alsugair

Spring 2022
Wednesday, March 2

Contents

Introduction

Dataset information for this project is obtained from UCI Machine Learning repository. The data includes training and testing data for classifying a high resolution aerial image into 9 types of urban land cover which are trees, grass, soil, concrete, asphalt, buildings, cars, pools, shadows. Features such as Multi-scale spectral, size, shape, and texture information are used for classification. There are a low number of training samples for each class (14-30) and a high number of classification variables or features (148 features in total). The testing data set is from a random sampling of the image. As the dataset contains a large number of features, it might be interesting to see their correlations between each pair of these variables in order to filter out some of them that have high correlation and use the remaining feature for prediction.

Details of Attributes

- Class : Land cover class (nominal)
- BrdIndx : Border Index (shape variable)
- Area : Area in m2 (size variable)
- Round : Roundness (shape variable)
- Bright : Brightness (spectral variable)
- Compact : Compactness (shape variable)
- ShpIndx : Shape Index (shape variable)
- Mean_G : Green (spectral variable)
- Mean_R : Red (spectral variable)
- Mean_NIR : Near Infrared (spectral variable)
- SD_G : Standard deviation of Green (texture variable)
- SD_R : Standard deviation of Red (texture variable)
- SD_NIR : Standard deviation of Near Infrared (texture variable)
- LW : Length/Width (shape variable)
- GLCM1 : Gray-Level Co-occurrence Matrix (texture variable)
- Rect : Rectangularity (shape variable)
- GLCM2 : Another Gray-Level Co-occurrence Matrix attribute (texture variable)
- Dens : Density (shape variable)
- Assym : Assymetry (shape variable)

- NDVI : Normalized Difference Vegetation Index (spectral variable)
- BordLngth: Border Length (shape variable)
- GLCM3 : Another Gray-Level Co-occurrence Matrix attribute (texture variable)
- Note: These variables repeat for each coarser scale (i.e. variable_40, variable_60, . . . variable_140).

Loading Data

```
#load in training and test data
train = pd.read_csv('training.csv')
test = pd.read_csv('testing.csv')
print("Rows and Columns(Train): ",train.shape)
```

```
## Rows and Columns(Train):  (168, 148)
```

```
print("Rows and Columns(Test) : ",test.shape)
```

```
## Rows and Columns(Test) :  (507, 148)
```

Removing data with missing value

```
# check for missing values although it is clear there are none
train.isnull().any().any()
```

```
## False
```

```
# duplicated function of pandas returns a duplicate row as true and others as false
sum(train.duplicated())
```

```
## 0
```

Basic Statistics

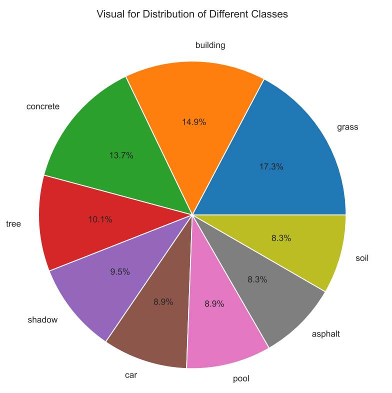
```
# basic statistical details
fig = train.describe().T
fig = fig.round(5)  # round to 5 decimal places
table = go.Table(
    columnwidth=[0.8]+[0.5]*8,
    header=dict(
        values=['Attribute'] + list(fig.columns),
        line = dict(color='darkslategray'),
        fill = dict(color='royalblue'),
    ),
    cells=dict(
        values=[fig.index] + [fig[k].tolist() for k in fig.columns[:]],
        line = dict(color='darkslategray'),
        fill = dict(color=['paleturquoise', 'white'])
    )
)
plot([table], filename='table-of-data.html')
```

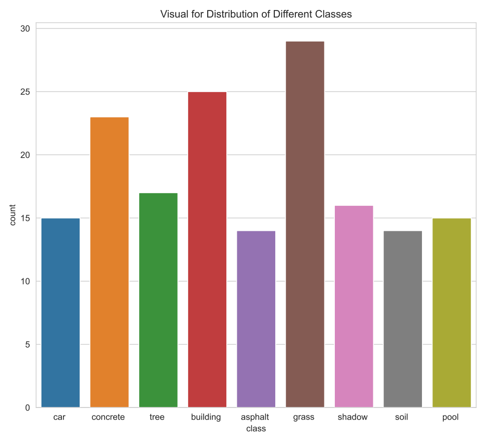
```
## 'table-of-data.html'
```

```
# more general data exploration
print(train['class'].value_counts())
```

```
## grass        29
## building     25
## concrete     23
## tree         17
## shadow       16
## car          15
## pool         15
## asphalt      14
## soil         14
## Name: class, dtype: int64
```

```
f,axes=plt.subplots(1,2,figsize=(20,8))
train['class'].value_counts().plot.pie(autopct='%1.1f%%',ax=axes[0])
axes[0].set_title('Visual for Distribution of Different Classes')
axes[0].set_ylabel('')
sns.countplot('class',data=train,ax=axes[1]) # sns.countplot is used
                                             # like a histogram but for
                                             # categorical data
axes[1].set_title('Visual for Distribution of Different Classes')
plt.show()
```





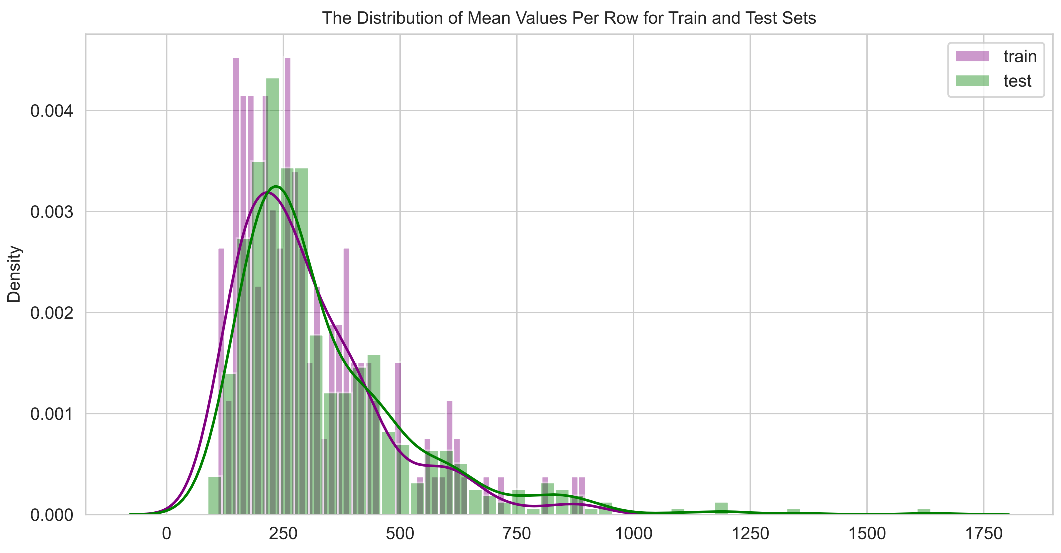
Outlier examination

```
# Lets take a look at any outliers that could be potential issues
from collections import Counter
def examine_outliers(train_data, n, features):
  outlier_indicator = []
  for out in features:
    Q1 = np.percentile(train_data[out], 25)
    Q3 = np.percentile(train_data[out], 75)
    IQR = Q3 - Q1
    outlier_step = 1.5 * IQR # IQR method of dealing with outliers, 1 of 2 methods
    outlier_list_out = train_data[
            (train_data[out] < Q1 - outlier_step) | (train_data[out] > Q3 +
            outlier_step)].index
    outlier_indicator.extend(outlier_list_out)
    outlier_indices = Counter(outlier_indicator)
    multiple_outliers = list(k for k, j in outlier_indices.items() if j > n)
    return multiple_outliers
# find outliers that should be removed
list_atributes = train.drop('class', axis=1).columns
outliers_to_remove = examine_outliers(train, 2, list_atributes)
len(outliers_to_remove)
#outliers_to_remove
#train.loc[outliers_to_remove]
```

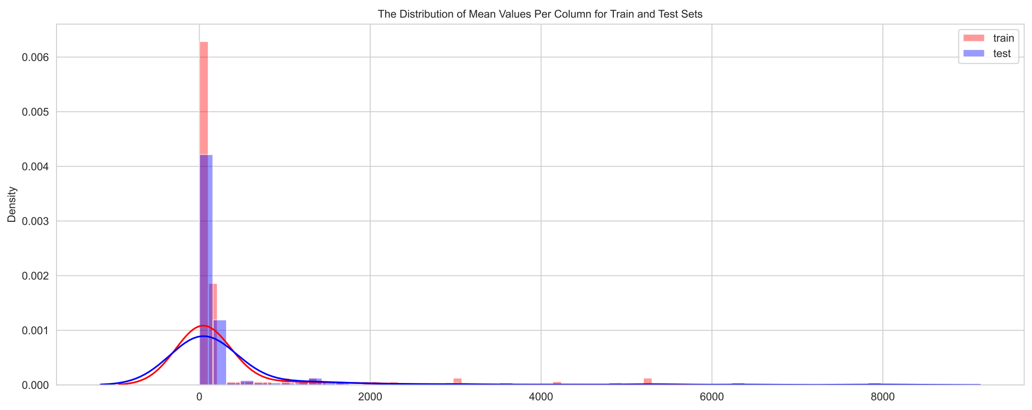
```
## 0
```

Distribution of Mean values

```
# lets look at mean values per row and column for train test data
# mean for rows
plt.figure(figsize=(10,5))
features = train.columns.values[1:148]
plt.title("The Distribution of Mean Values Per Row for Train and Test Sets",
          fontsize=10)
sns.distplot(train[features].mean(axis=1),color="purple", kde=True,bins=50,
            label='train') # kde is kernel density estimation
sns.distplot(test[features].mean(axis=1),color="green", kde=True,bins=50,
            label='test')
plt.legend()
plt.show()
```



```
# mean for columns
plt.figure(figsize=(16,6))
plt.title("The Distribution of Mean Values Per Column for Train and Test Sets",
         fontsize=10)
sns.distplot(train[features].mean(axis=0),color="red",kde=True,bins=50,
             label='train')
sns.distplot(test[features].mean(axis=0),color="blue", kde=True,bins=50,
            label='test')
plt.legend()
plt.show()
```



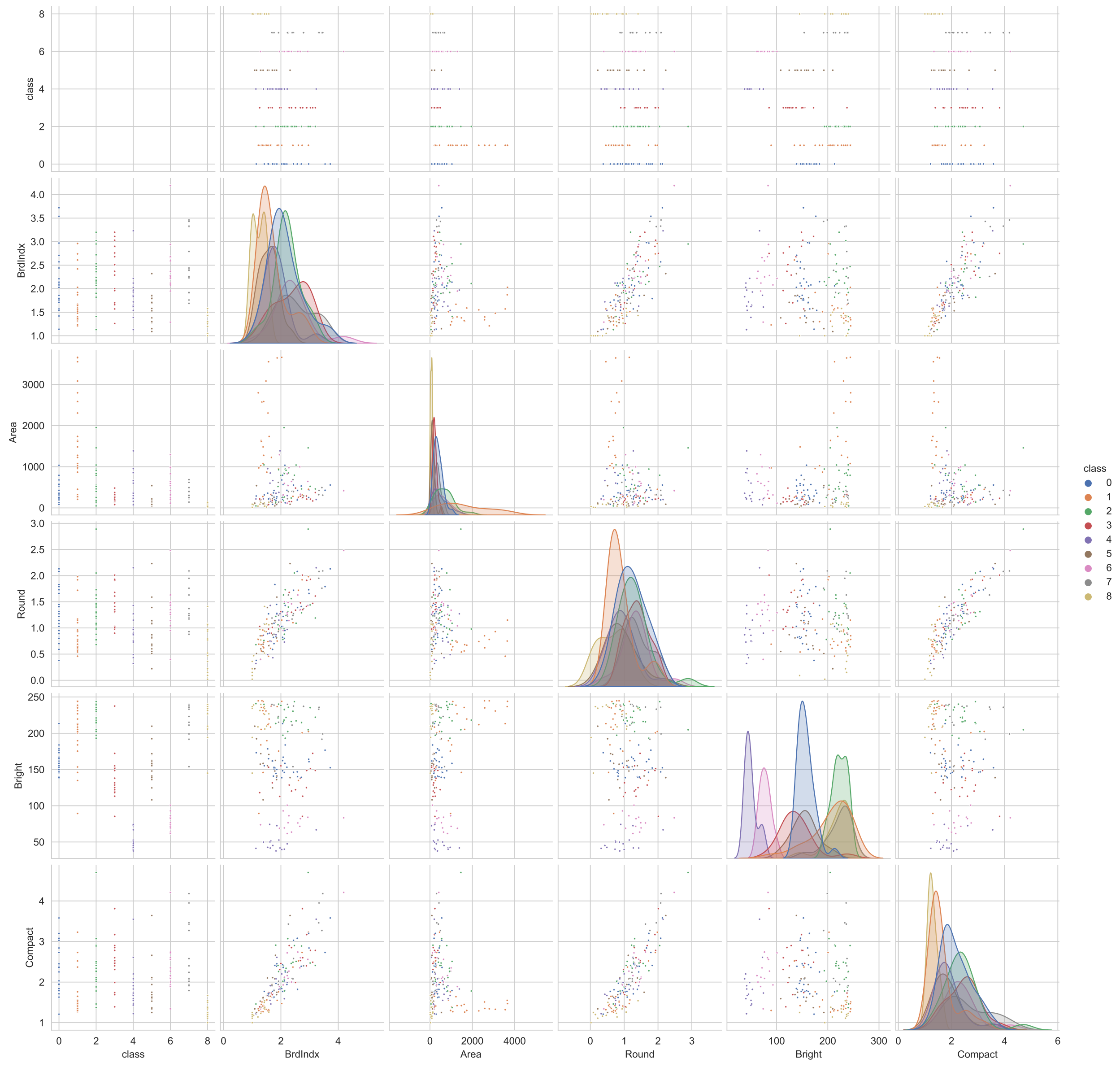
Attributes Correlation

```
# lets examine correlations between features
# first the categorical variable 'class' needs to be changed to numerical variable so...
```

```
group_map = {"grass ":0,"building ":1,'concrete ':2,'tree ':3,'shadow ':4,'pool ':5,
             'asphalt ':6,'soil ':7,'car ':8}
train['class'] = train['class'].map(group_map)
test['class'] = test['class'].map(group_map)
train['class'].unique()
```

```
## array([8, 2, 3, 1, 6, 0, 4, 7, 5], dtype=int64)
```

```
# now lets look at the correlation between a few variables
sns.pairplot(train, vars=['class', 'BrdIndx','Area','Round','Bright','Compact'],
             hue='class', palette='deep', plot_kws={"s": 4})
#plt.show()
```



```
# correlation of features with target
corr = train.corr().abs().unstack().sort_values(kind="quicksort").reset_index()
corr = corr[corr['level_0'] != corr['level_1']]
corr.head()
```

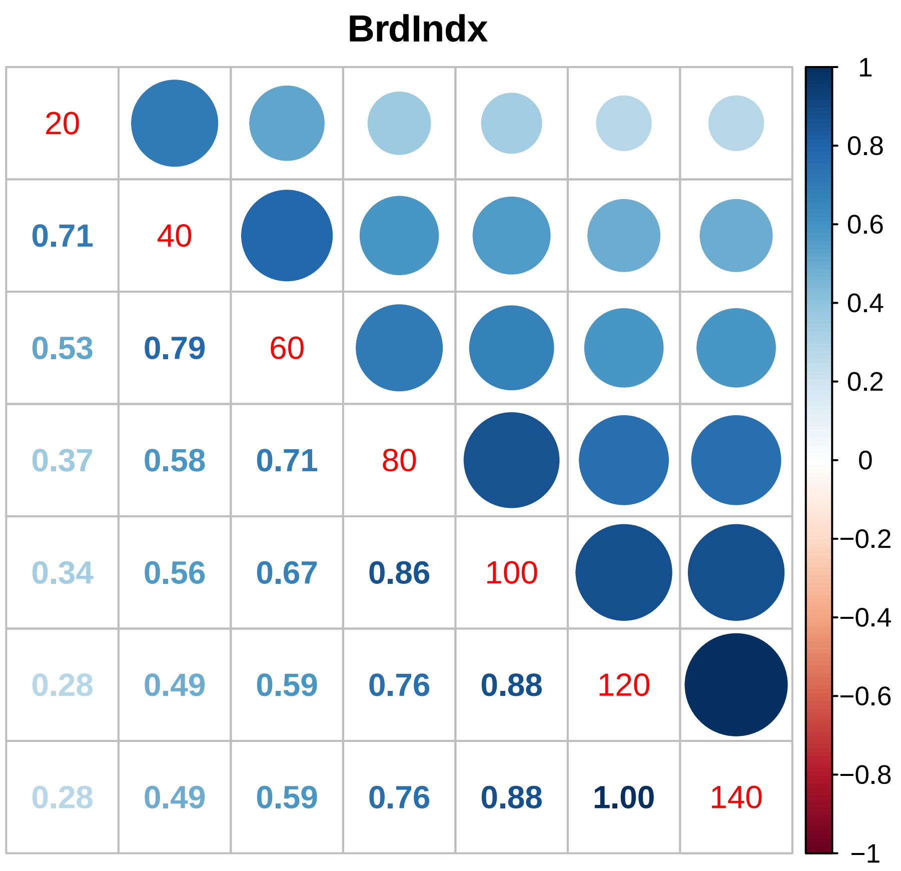
```
##      level_0    level_1         0
## 0       SD_R   GLCM1_60  0.000052
## 1   GLCM1_60       SD_R  0.000052
## 2   Mean_NIR  BordLngth  0.000162
## 3  BordLngth   Mean_NIR  0.000162
## 4  Mean_R_40   Round_40  0.000190
```

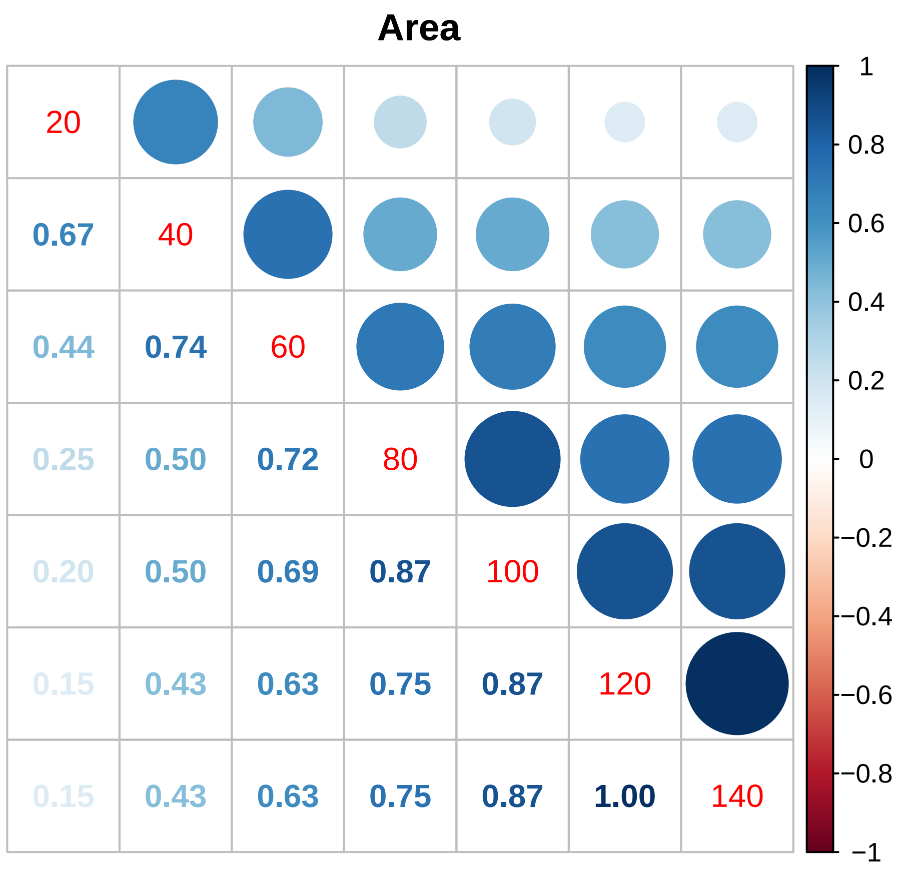
```
correlations = corr.loc[corr[0] == 1]
features_to_be_removed = set(list(correlations['level_1']))
correlations.shape
```

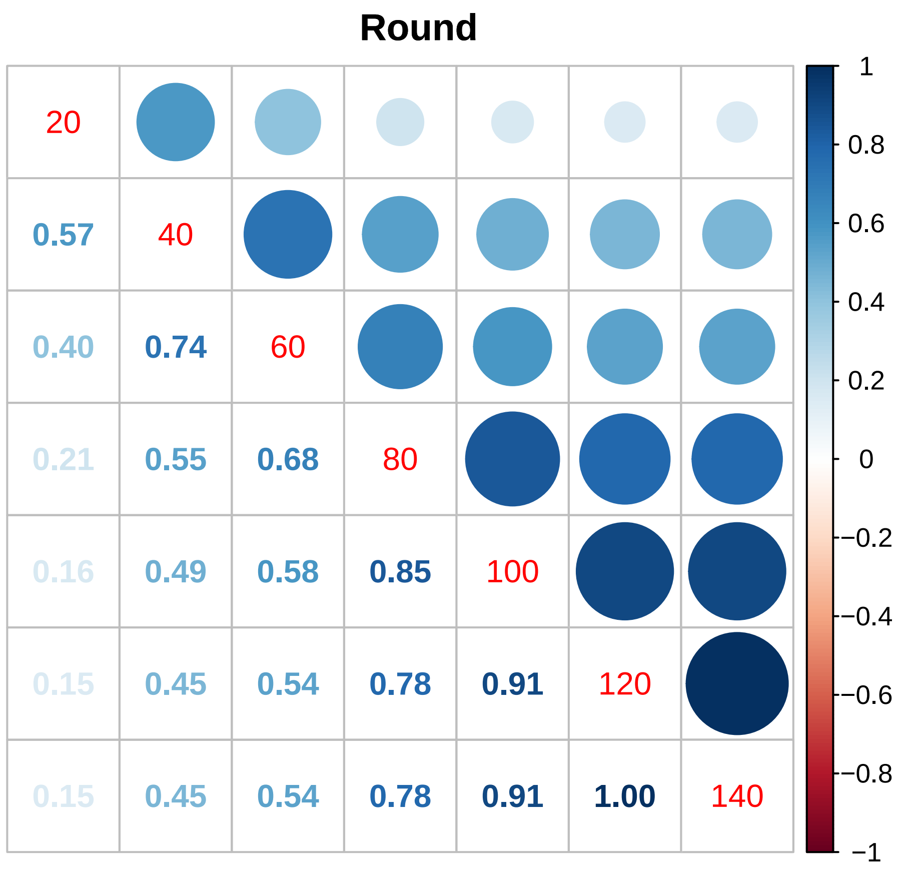
```
## (42, 3)
```

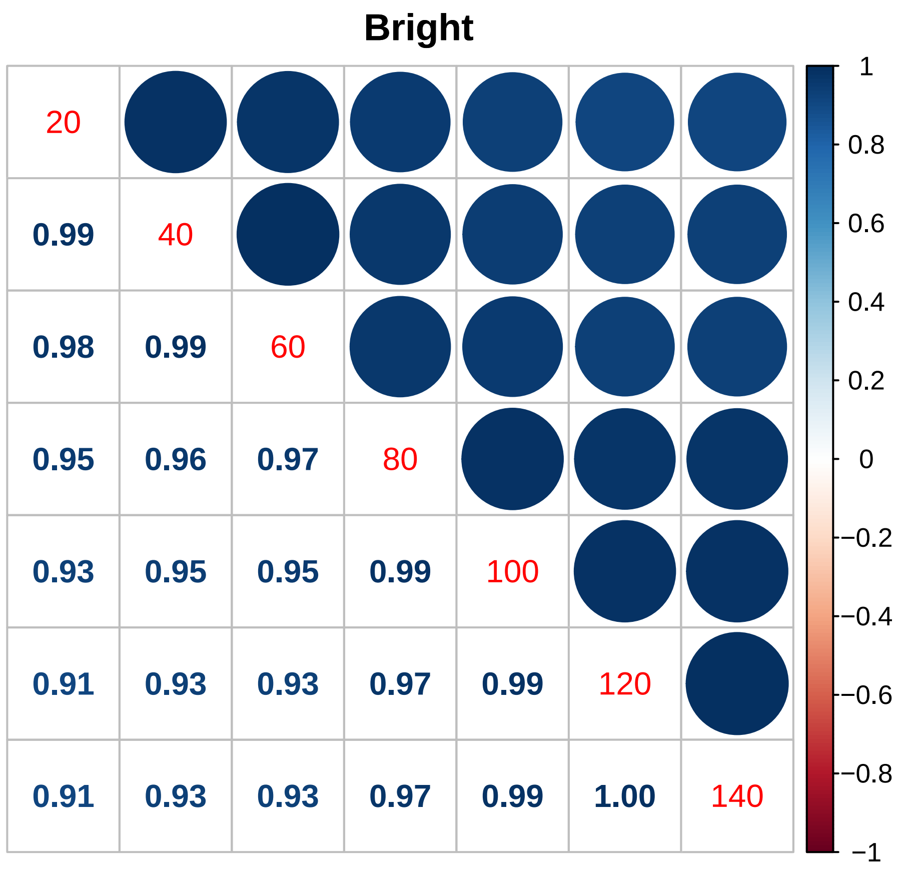
```
# A first start would be to investigate each coarseness level
#create a list of dataframes
trainingList <- list()

#create new data-frame for each coarseness level

for( i in 1:7){
  #set the name to data + coarseness value
  name<- paste("data", i*20, sep = "")
  #first column index of the selected coarseness
  begin = (i-1)*21+2
  #last column index
  ending = begin+20
  #assign the first column and the columns between first and last
  assign(name, trainData[,c(1,begin:ending)])
  # add to the list
  trainingList[[i]] <- trainData[,c(1,begin:ending)]
}
```

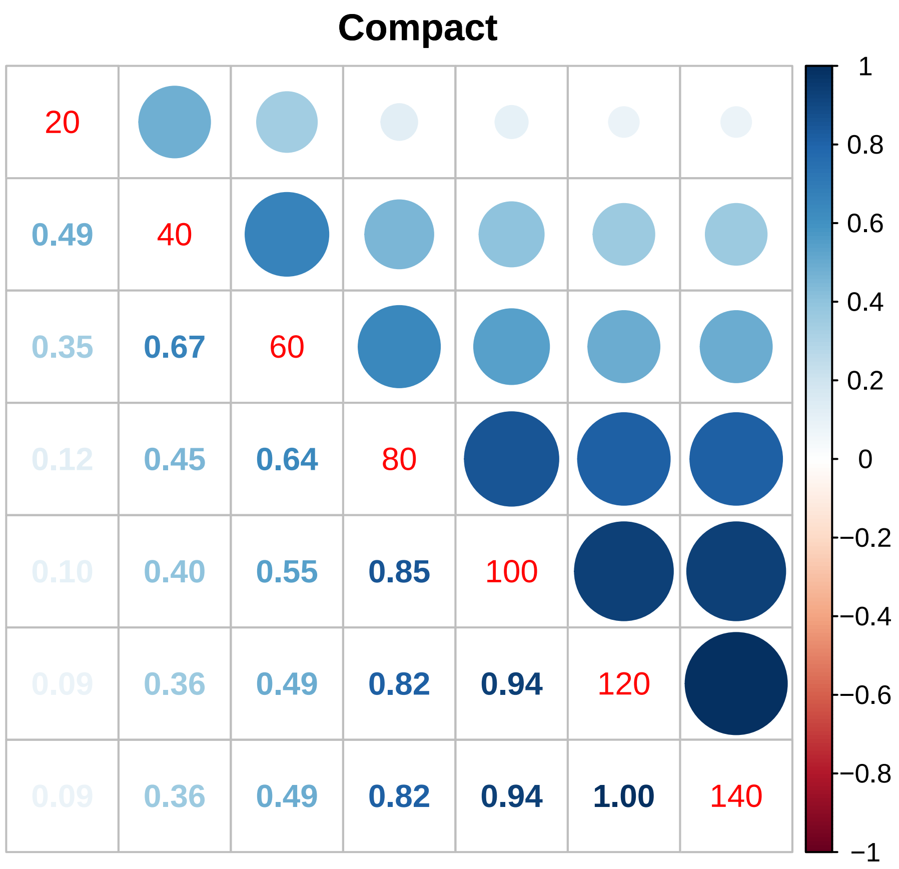
```
# check collinearity among repeated variables

corList <- list()
#looping through all repeated variables
for(i in 2:22){
  #creating a dataframe with coarseness 20, and attribute i
  df <- trainingList[[1]][,i]
  for(j in 2:7){
    #adding the same attribute from a different coarseness level
    df <- cbind(df,trainingList[[j]][,i] )
    colnames(df)[j]<- paste(j*20)
  }
  colnames(df)[1]<- "20"
  #adding a correlation matrix to the list
  corList[[i]] <-  cor(df)
  #plotting 21 correlation matrices
  corrplot.mixed(corList[[i]], upper = 'circle', lower = 'number', title =
                    colnames(data20[i]), mar=c(0,0,1,0))
}
```

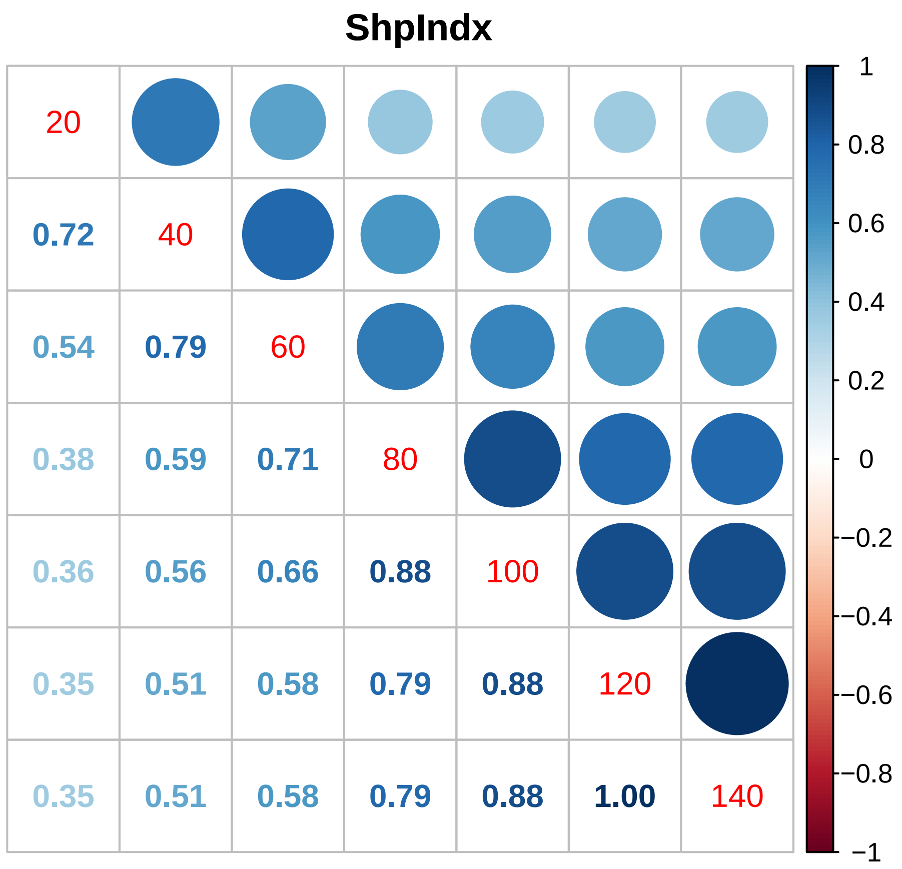



Area
20
40
60
80
100
120
140
0.67
0.44
0.74
0.25
0.50
0.72
0.20
0.50
0.69
0.87
0.15
0.43
0.63
0.75
0.87
0.15
0.43
0.63
0.75
0.87
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

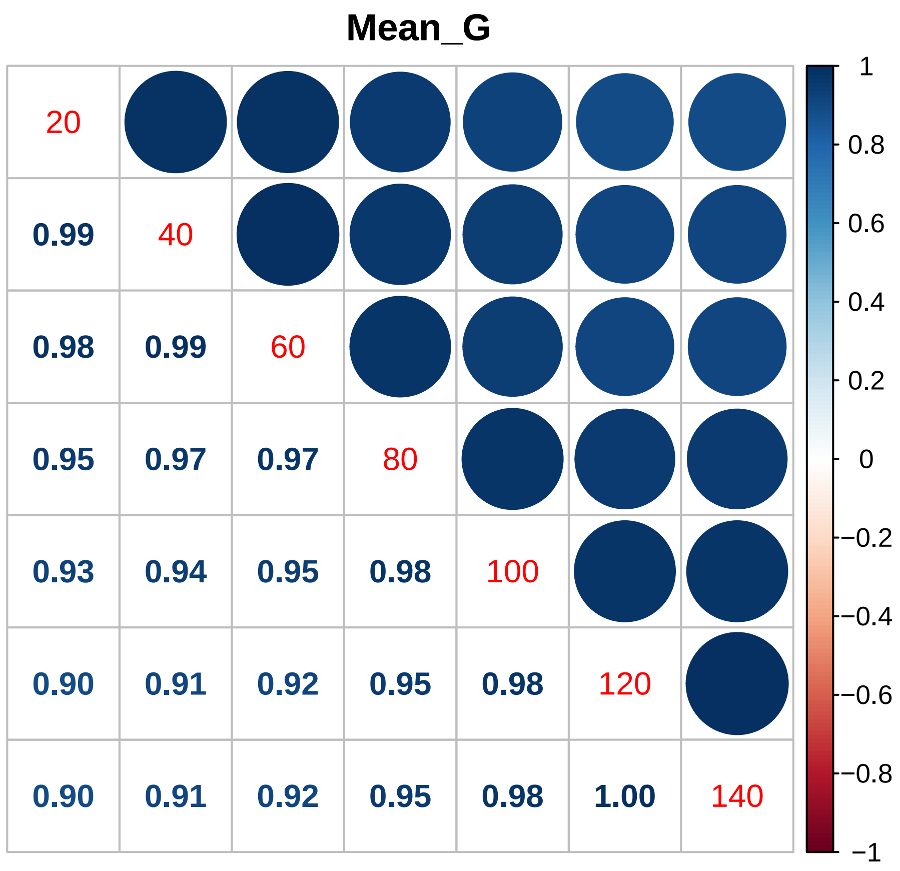

Round
20
40
60
80
100
120
140
0.57
0.40
0.74
0.21
0.55
0.68
0.16
0.49
0.58
0.85
0.15
0.45
0.54
0.78
0.91
0.15
0.45
0.54
0.78
0.91
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

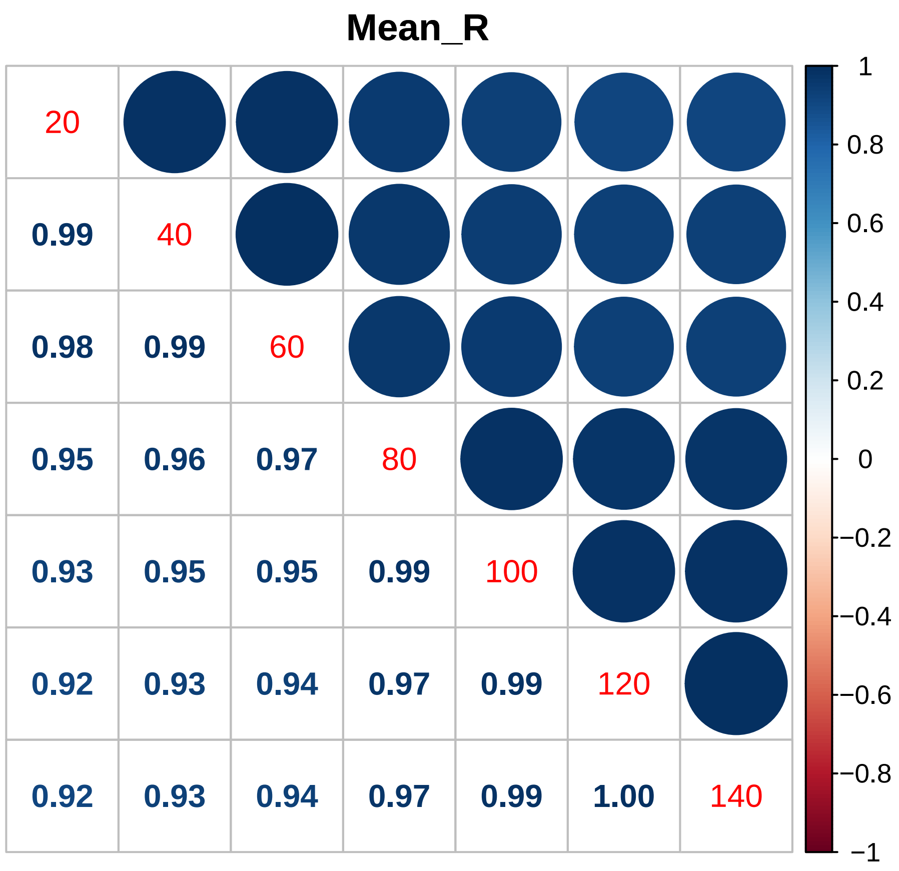

Bright
20
40
60
80
100
120
140
0.99
0.98
0.99
0.95
0.96
0.97
0.93
0.95
0.95
0.99
0.91
0.93
0.93
0.97
0.99
0.91
0.93
0.93
0.97
0.99
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

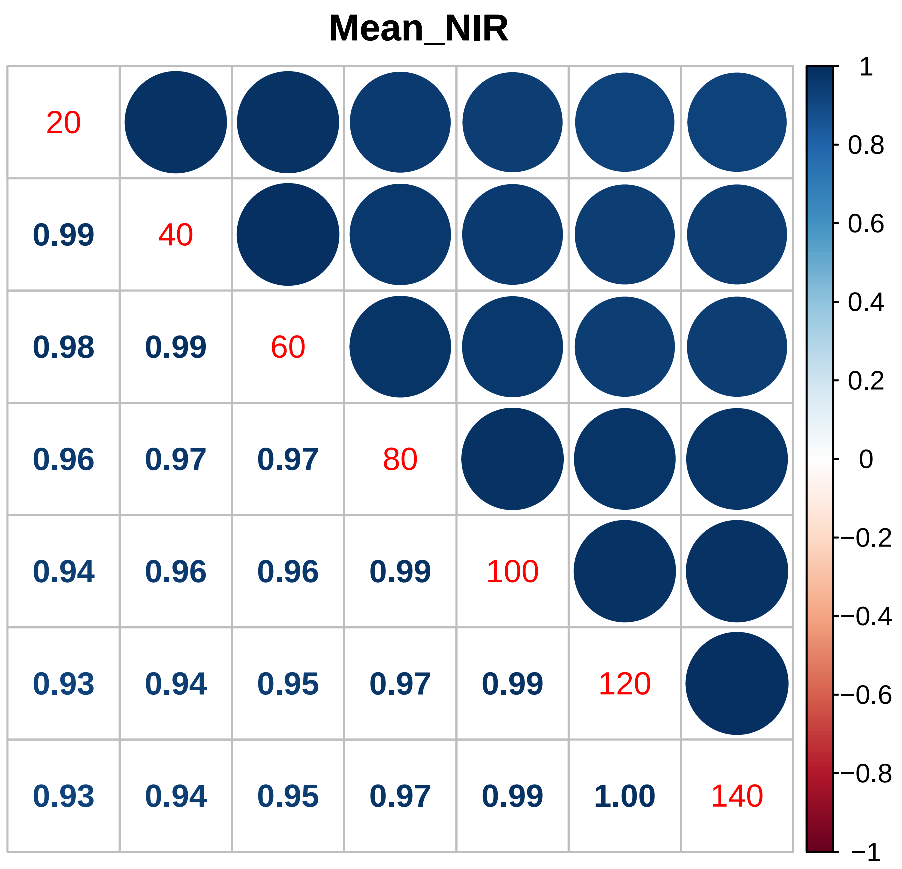

Compact
20
0.49
40
0.35
0.67
60
0.12
0.45
0.64
80
0.10
0.40
0.55
0.85
100
0.09
0.36
0.49
0.82
0.94
120
0.09
0.36
0.49
0.82
0.94
1.00
140
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

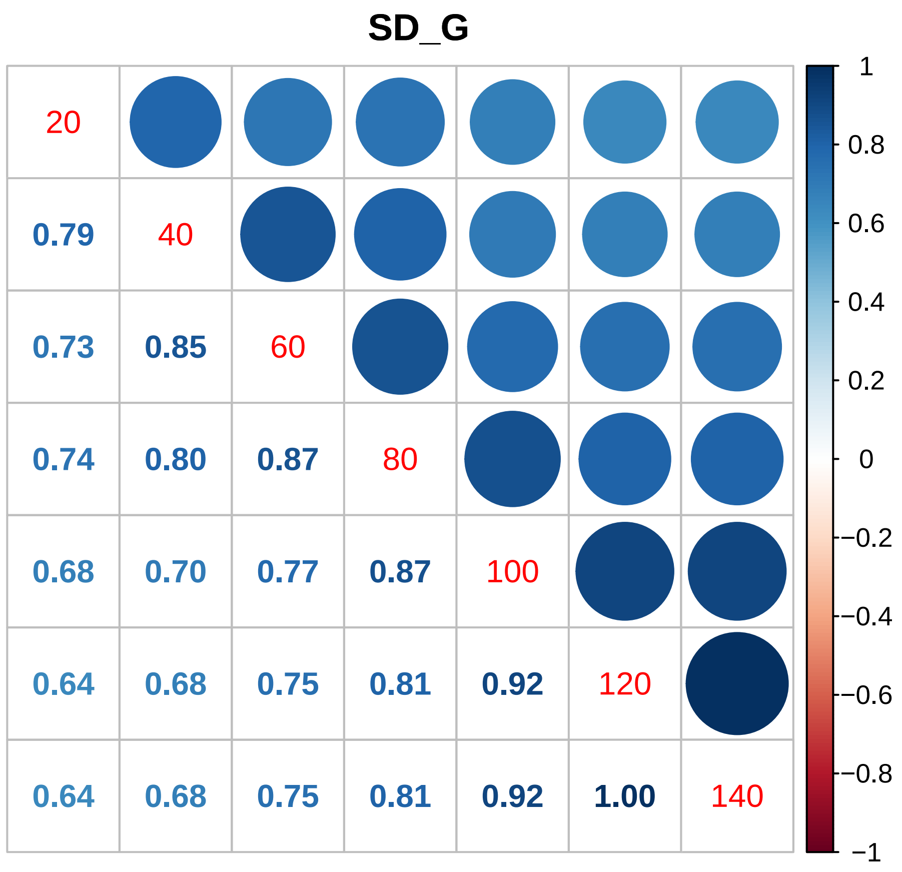


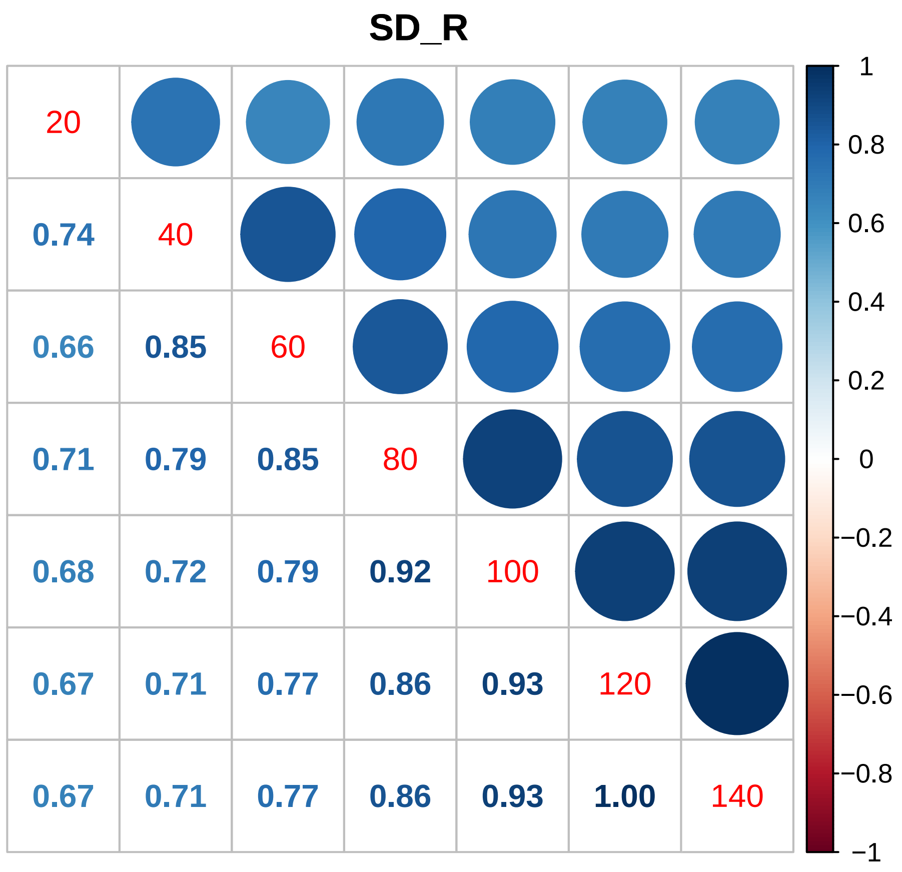
ShpIndx
20
40
60
80
100
120
140
0.72
0.54
0.79
0.38
0.59
0.71
0.36
0.56
0.66
0.88
0.35
0.51
0.58
0.79
0.88
0.35
0.51
0.58
0.79
0.88
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

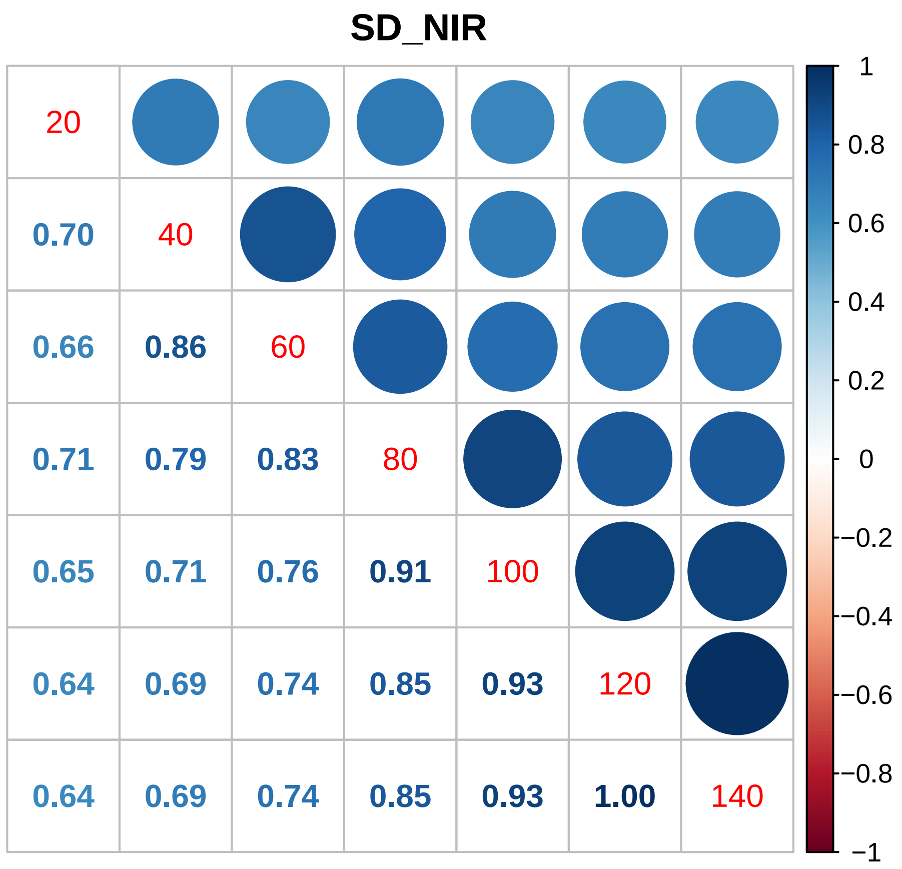

Mean_G
20
40
60
80
100
120
140
0.99
0.98
0.99
0.95
0.97
0.97
0.93
0.94
0.95
0.98
0.90
0.91
0.92
0.95
0.98
0.90
0.91
0.92
0.95
0.98
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

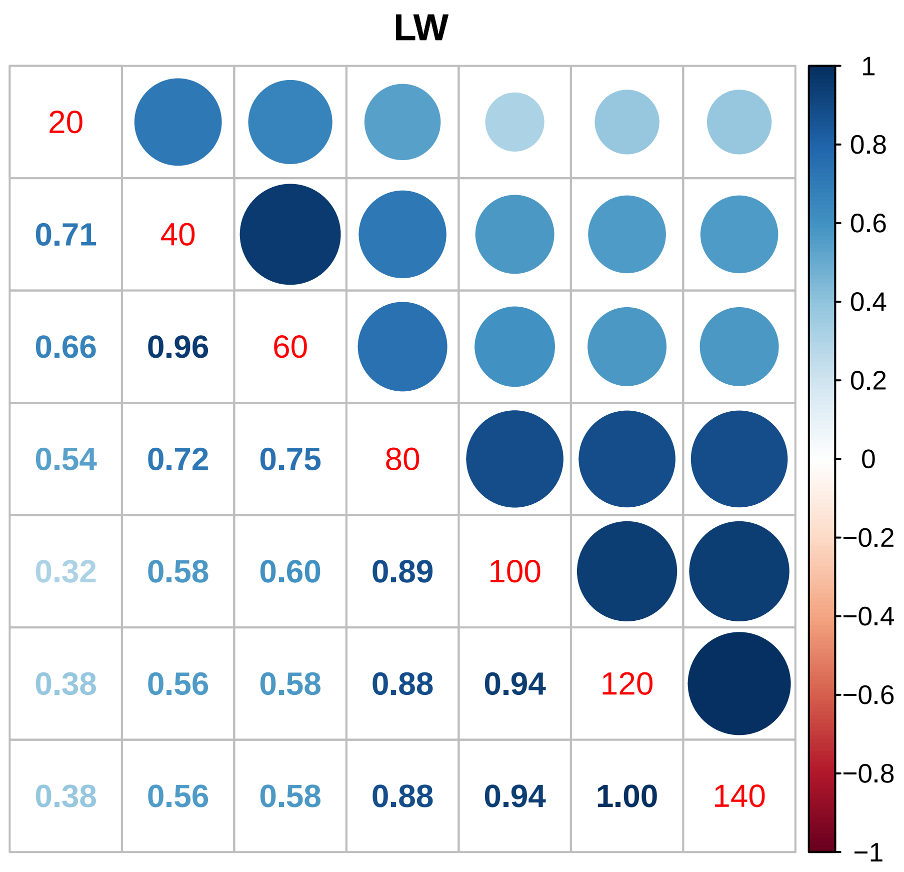

Mean_R
20
40
60
80
100
120
140
0.99
0.98 0.99
0.95 0.96 0.97
0.93 0.95 0.95 0.99
0.92 0.93 0.94 0.97 0.99
0.92 0.93 0.94 0.97 0.99 1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

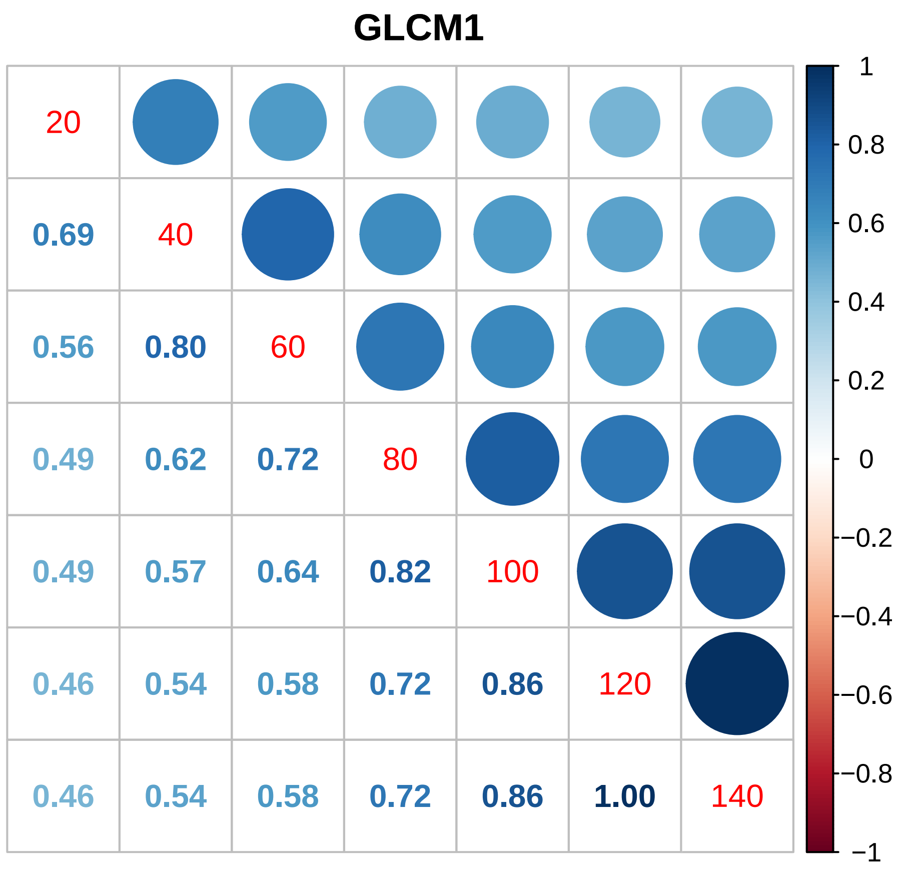

Mean_NIR
20
40
60
80
100
120
140
0.99
0.98
0.99
0.96
0.97
0.97
0.94
0.96
0.96
0.99
0.93
0.94
0.95
0.97
0.99
0.93
0.94
0.95
0.97
0.99
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

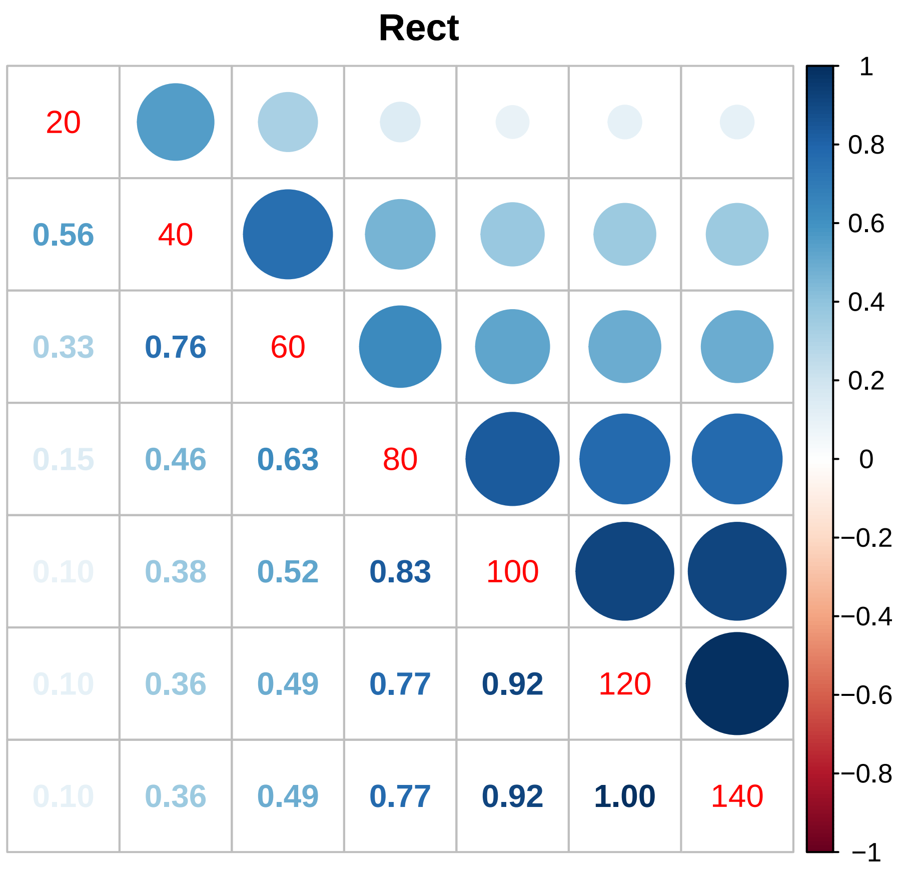

SD_G
20
40
60
80
100
120
140
0.79
0.73
0.85
0.74
0.80
0.87
0.68
0.70
0.77
0.87
0.64
0.68
0.75
0.81
0.92
0.64
0.68
0.75
0.81
0.92
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

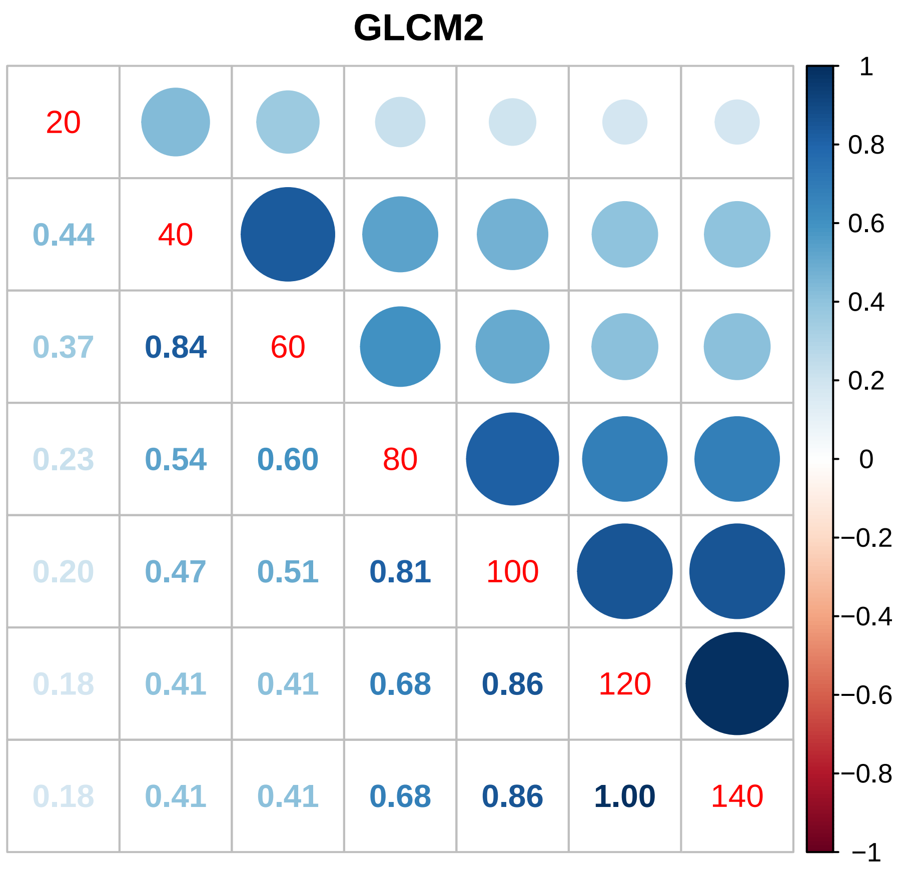

SD_R
20
40
60
80
100
120
140
0.74
0.66
0.85
0.71
0.79
0.85
0.68
0.72
0.79
0.92
0.67
0.71
0.77
0.86
0.93
0.67
0.71
0.77
0.86
0.93
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

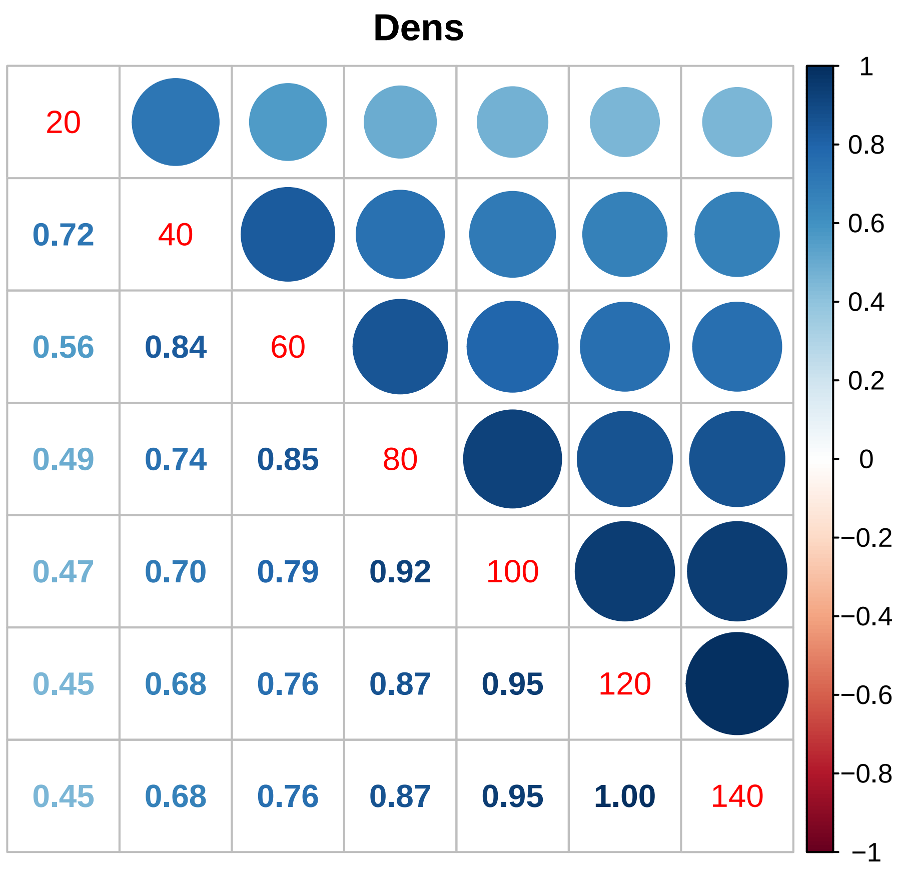

SD_NIR
20
40
60
80
100
120
140
0.70
0.66
0.86
0.71
0.79
0.83
0.65
0.71
0.76
0.91
0.64
0.69
0.74
0.85
0.93
0.64
0.69
0.74
0.85
0.93
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

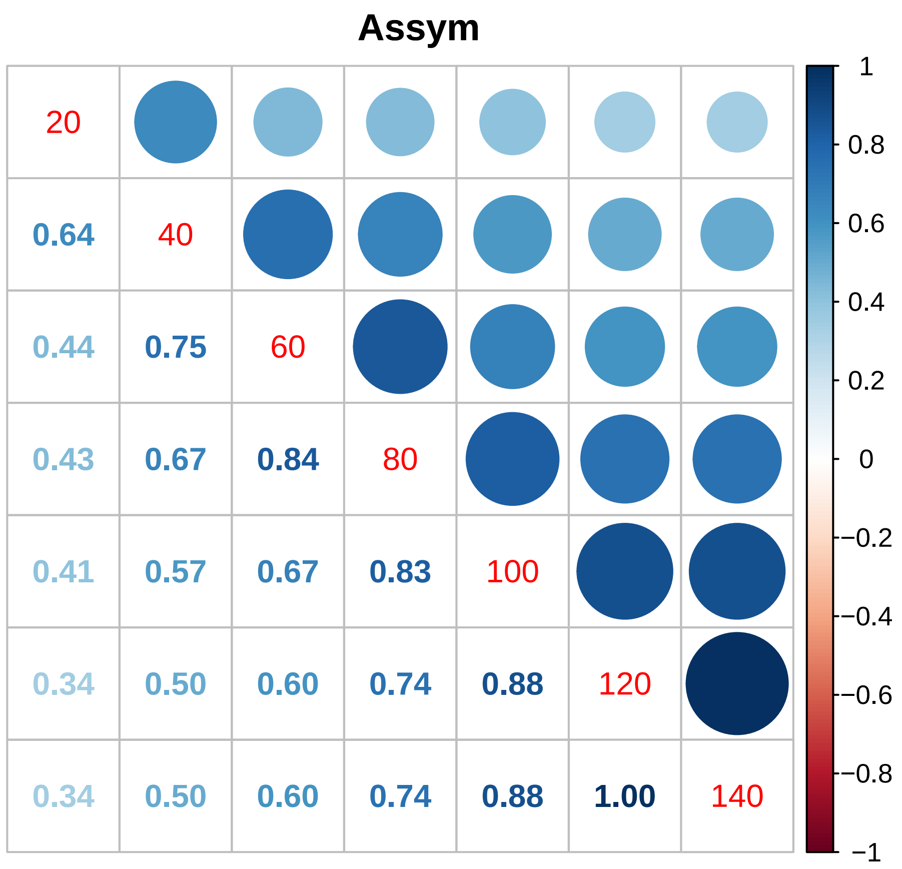

LW
20
40
60
80
100
120
140
0.71
0.66
0.96
0.54
0.72
0.75
0.32
0.58
0.60
0.89
0.38
0.56
0.58
0.88
0.94
0.38
0.56
0.58
0.88
0.94
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

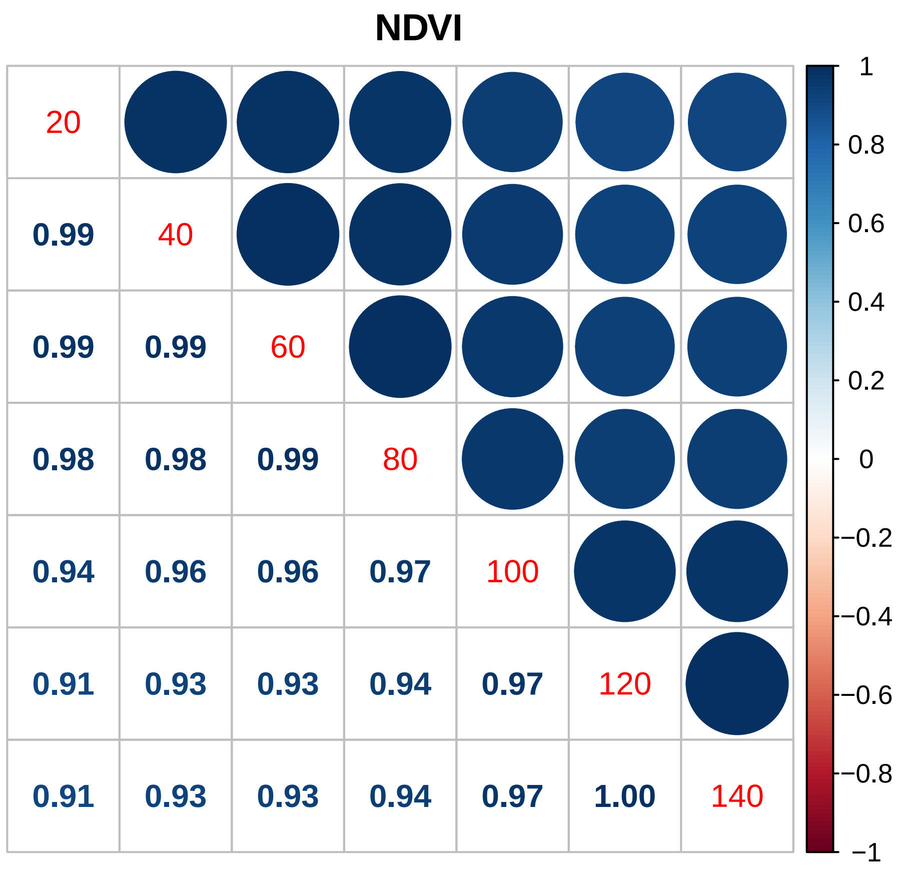

GLCM1
20
0.69 40
0.56 0.80 60
0.49 0.62 0.72 80
0.49 0.57 0.64 0.82 100
0.46 0.54 0.58 0.72 0.86 120
0.46 0.54 0.58 0.72 0.86 1.00 140
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

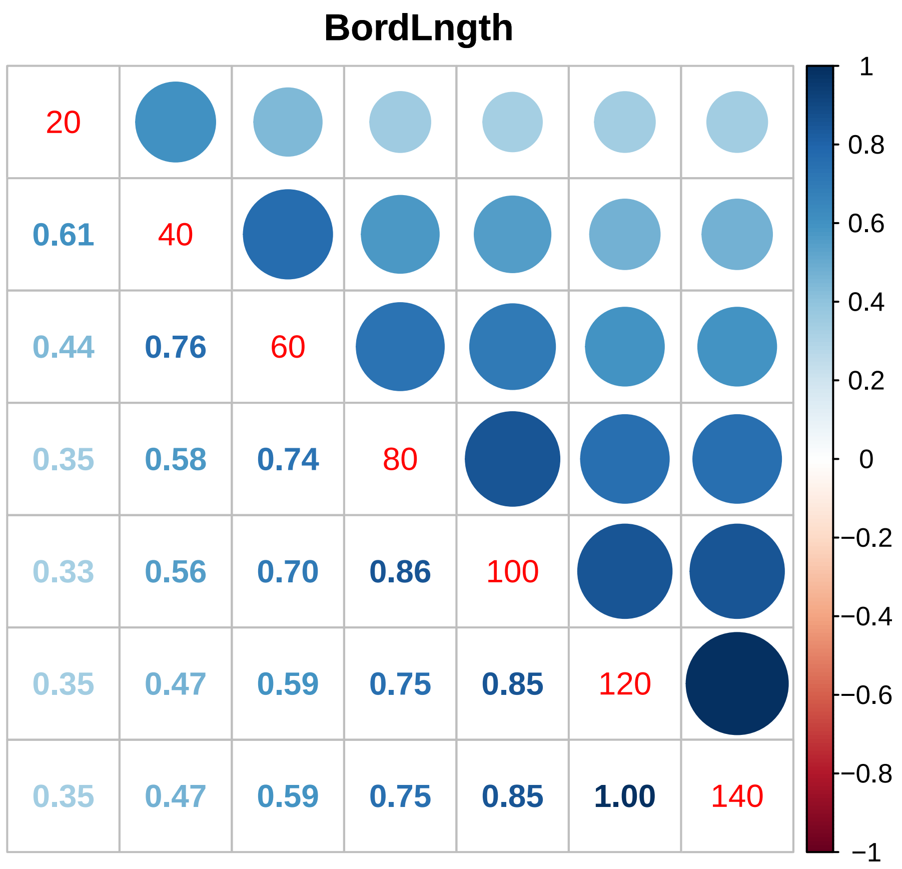

Rect
20
40
60
80
100
120
140
0.56
0.33
0.76
0.15
0.46
0.63
0.10
0.38
0.52
0.83
0.10
0.36
0.49
0.77
0.92
0.10
0.36
0.49
0.77
0.92
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

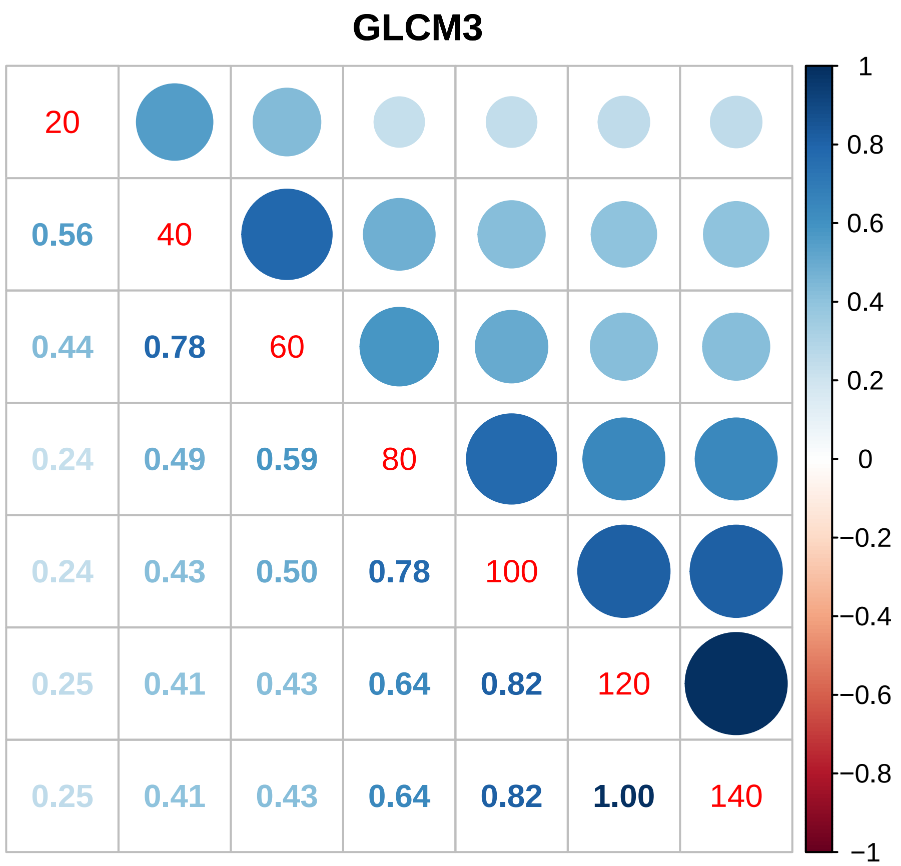

GLCM2
20
40
60
80
100
120
140
0.44
0.37
0.84
0.23
0.54
0.60
0.20
0.47
0.51
0.81
0.18
0.41
0.41
0.68
0.86
0.18
0.41
0.41
0.68
0.86
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1


Dens
20
40
60
80
100
120
140
0.72
0.56
0.84
0.49
0.74
0.85
0.47
0.70
0.79
0.92
0.45
0.68
0.76
0.87
0.95
0.45
0.68
0.76
0.87
0.95
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1


Assym
20
40
60
80
100
120
140
0.64
0.44 0.75
0.43 0.67 0.84
0.41 0.57 0.67 0.83
0.34 0.50 0.60 0.74 0.88
0.34 0.50 0.60 0.74 0.88 1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1


NDVI
20
40
60
80
100
120
140
0.99
0.99 0.99
0.98 0.98 0.99
0.94 0.96 0.96 0.97
0.91 0.93 0.93 0.94 0.97
0.91 0.93 0.93 0.94 0.97 1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

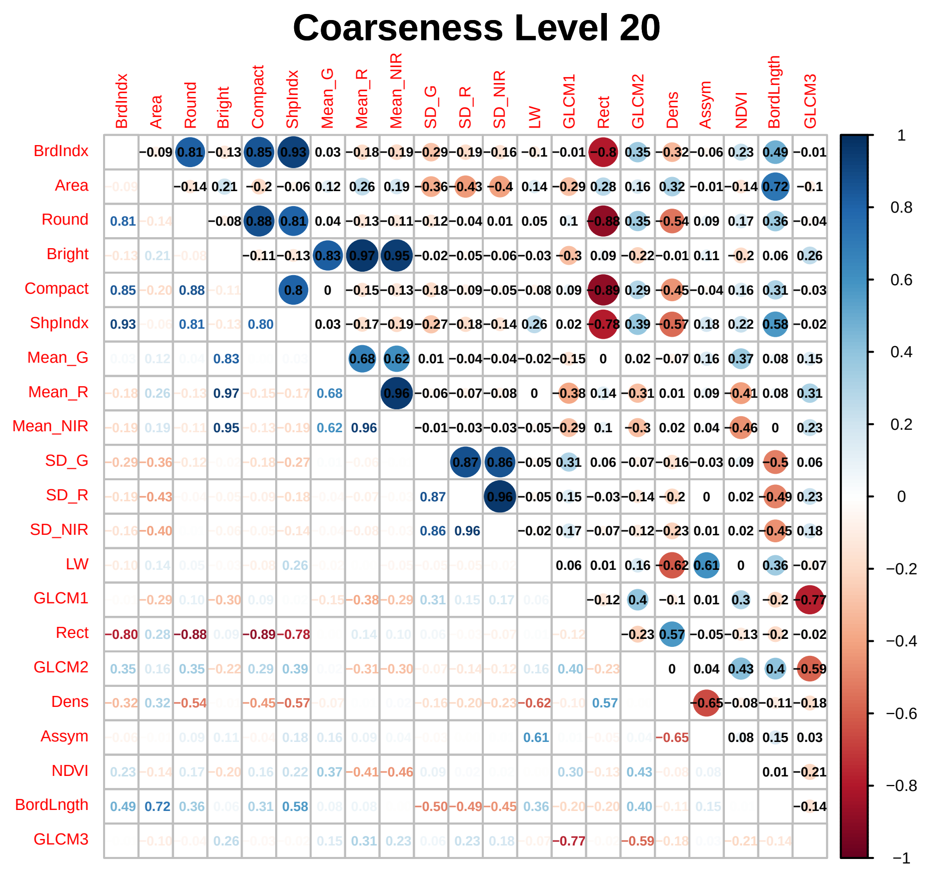

BordLngth
20
40
60
80
100
120
140
0.61
0.44
0.76
0.35
0.58
0.74
0.33
0.56
0.70
0.86
0.35
0.47
0.59
0.75
0.85
0.35
0.47
0.59
0.75
0.85
1.00
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

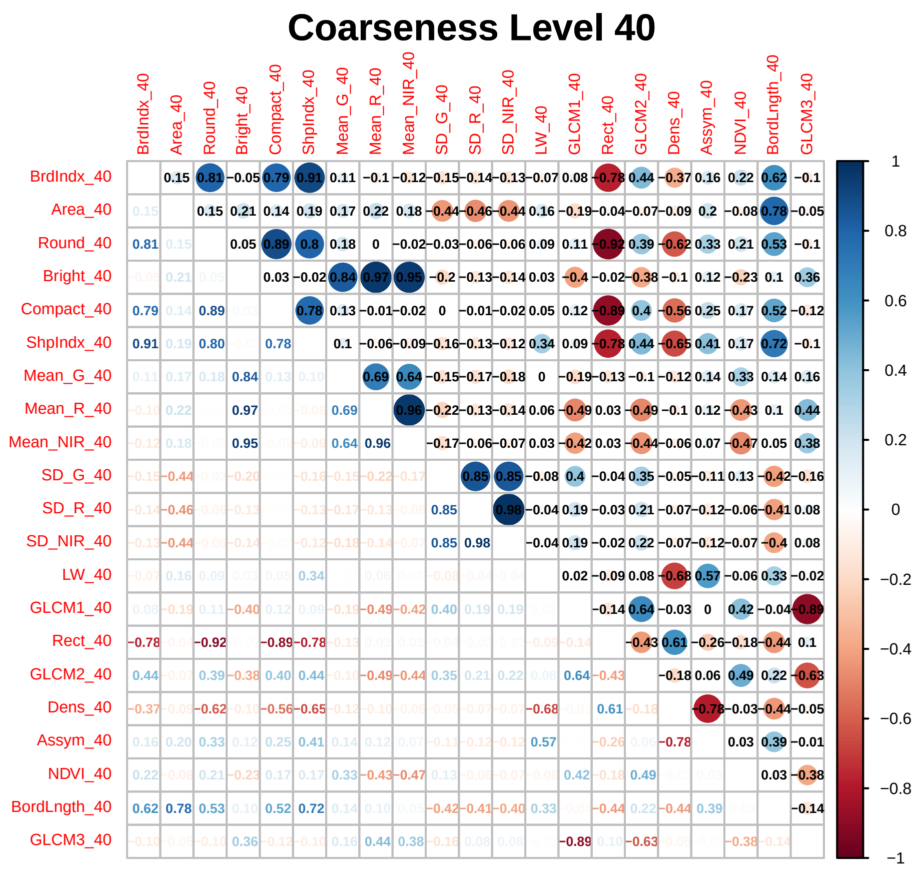


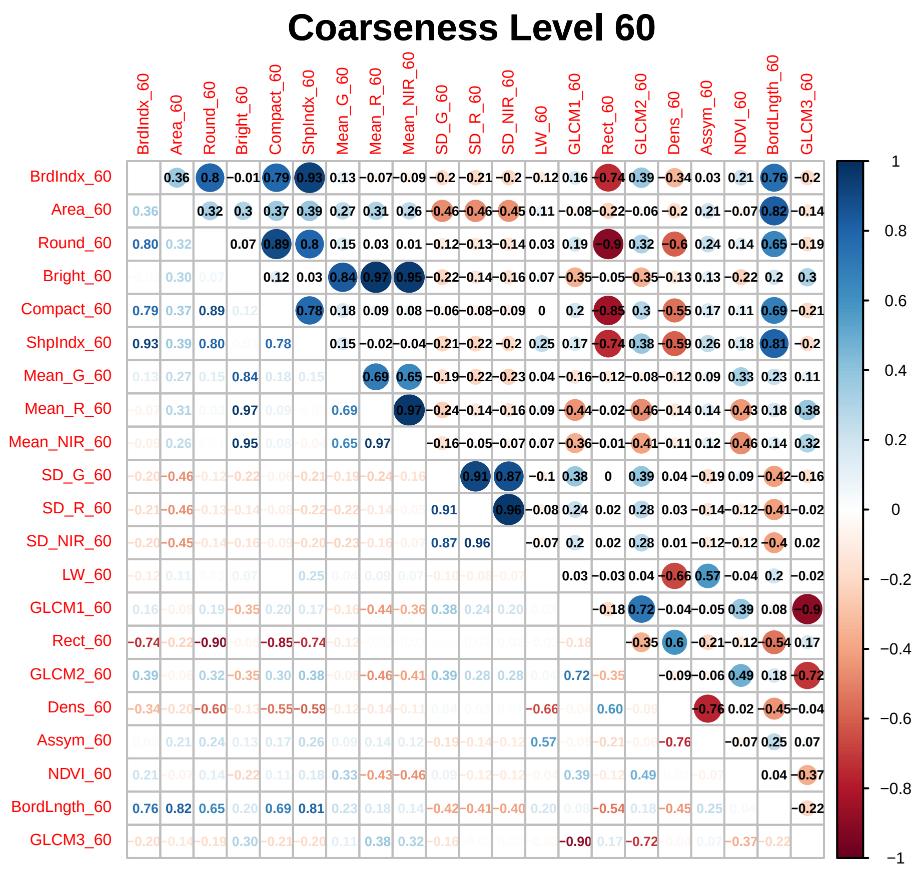
From the correlation plots: 1. In all coarseness cases, 120 and 140 are very highly correlated 2. across all attributes, correlation among consecutive coarseness level is highly correlated 3. Bright, Mean_G, Mean_R, Mean_NIR, and NVDI are extremely correlated across all coarseness levels

Hypothesis: perhaps modelling the data using a single coarseness level is sufficient or using coarseness levels 20 and 140 as they are they least correlated

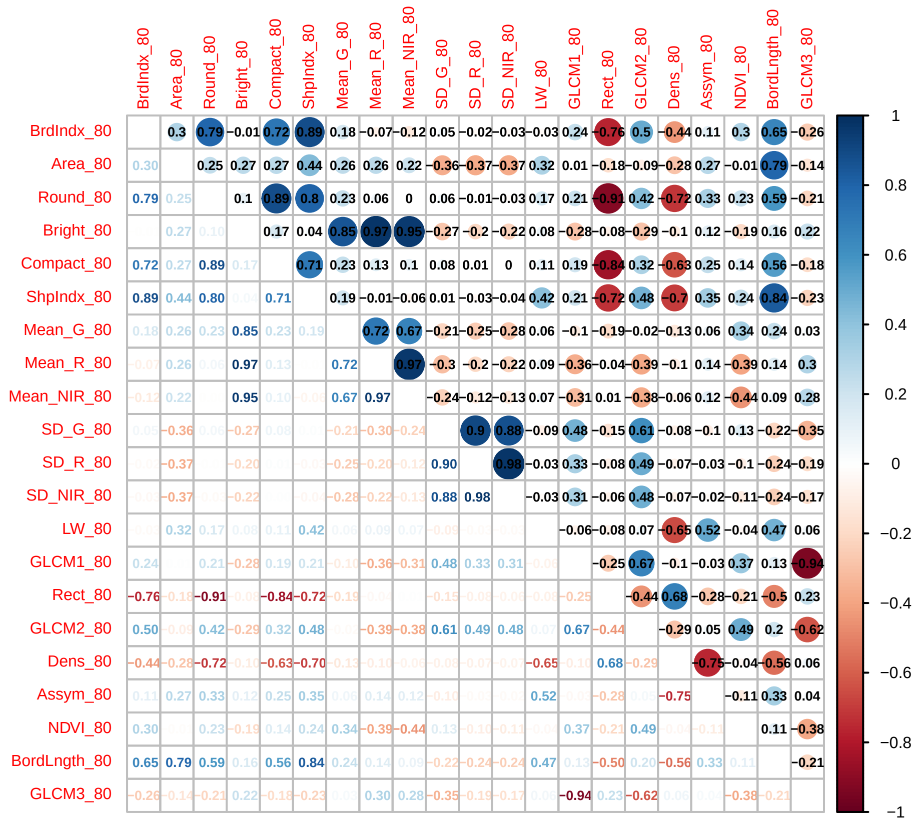
```
# checking the correlation among different attributes within the same coarseness


corList2 <- list()
for(i in 1:7){
  corList2[[i]] <- cor(trainingList[[i]][,2:22])
  corrplot.mixed(corList2[[i]],addCoef.col = 'black', number.cex = 0.4, upper = 'circle',
                 tl.cex=0.5, lower = 'number', title = paste("Coarseness Level",
                                                     i*20, sep=" "), mar=c(0,0,1,0),
                 cl.cex = 0.5, diag = "n", tl.pos = "lt")
}
```

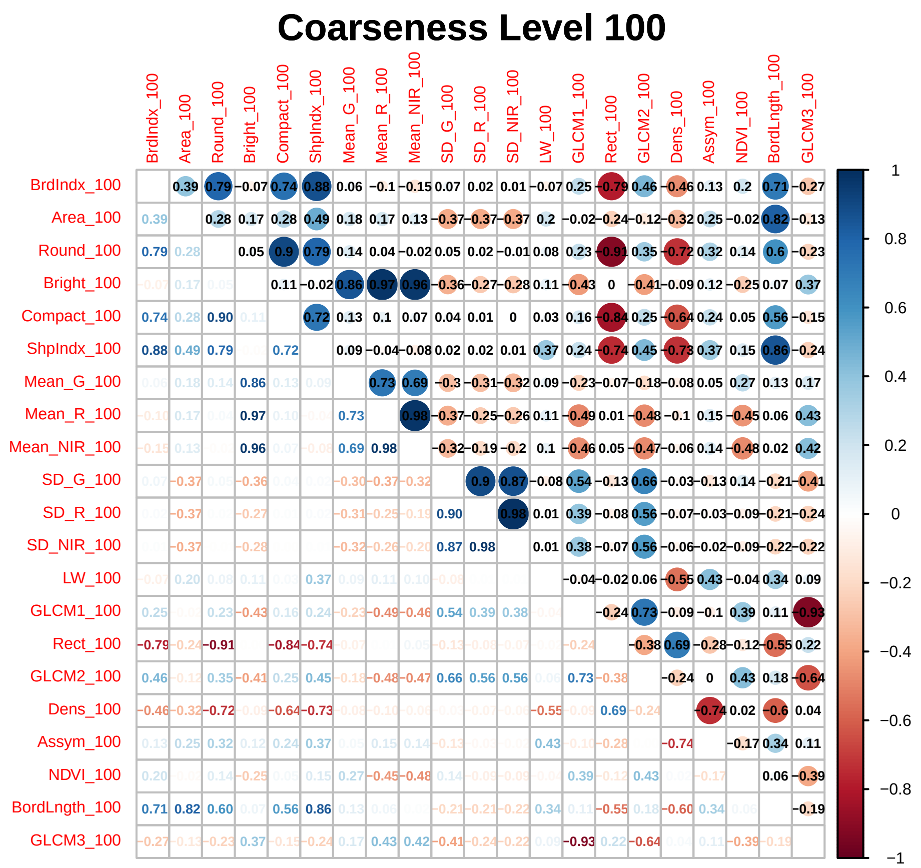

Coarseness Level 20
BrdIndx
Area
Round
Bright
Compact
ShpIndx
Mean_G
Mean_R
Mean_NIR
SD_G
SD_R
SD_NIR
LW
GLCM1
Rect
GLCM2
Dens
Assym
NDVI
BordLngth
GLCM3
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

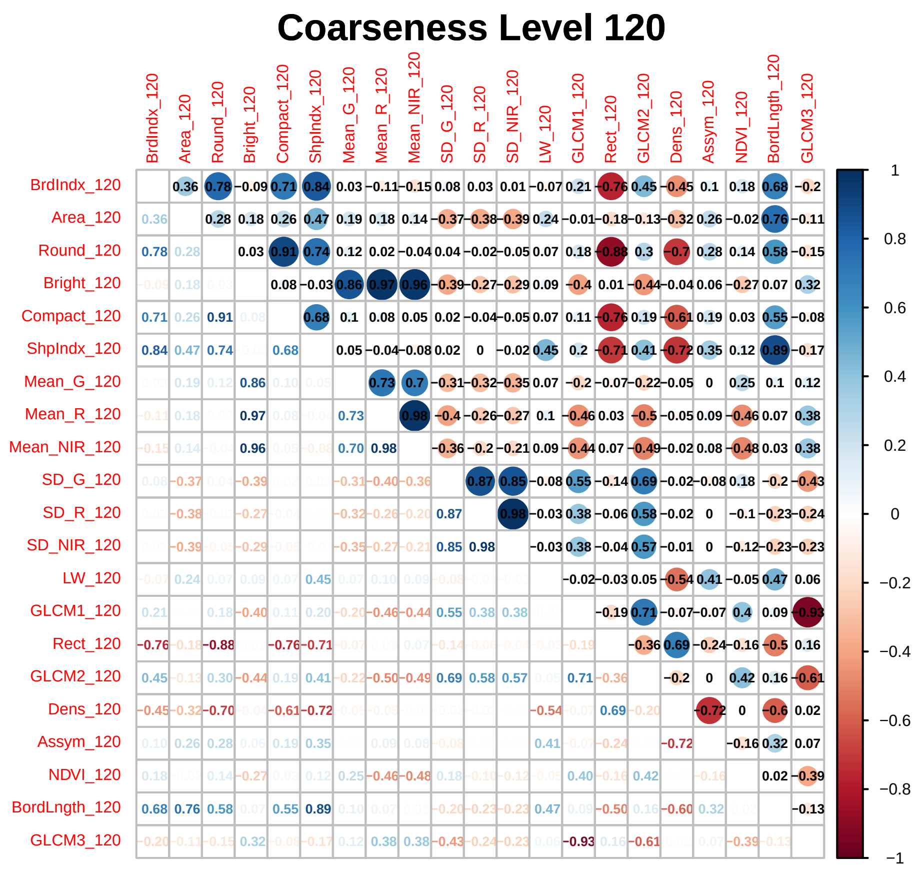

Coarseness Level 40
BrdIndx_40
Area_40
Round_40
Bright_40
Compact_40
ShpIndx_40
Mean_G_40
Mean_R_40
Mean_NIR_40
SD_G_40
SD_R_40
SD_NIR_40
LW_40
GLCM1_40
Rect_40
GLCM2_40
Dens_40
Assym_40
NDVI_40
BordLngth_40
GLCM3_40
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

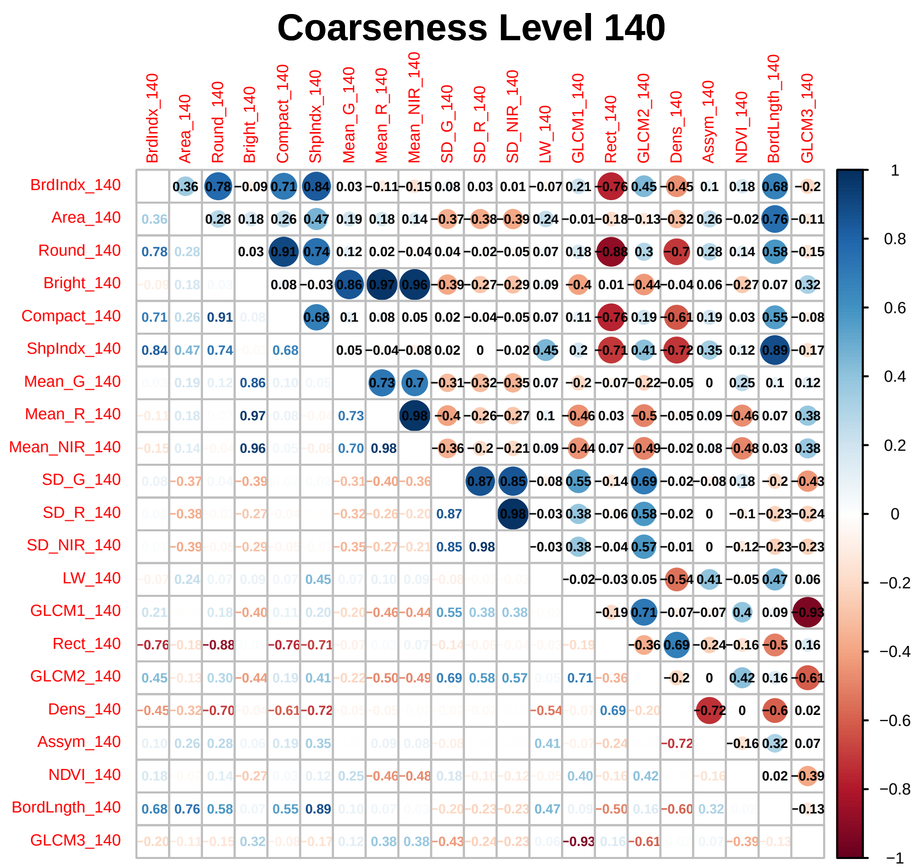

Coarseness Level 60
BrdIndx_60
Area_60
Round_60
Bright_60
Compact_60
ShpIndx_60
Mean_G_60
Mean_R_60
Mean_NIR_60
SD_G_60
SD_R_60
SD_NIR_60
LW_60
GLCM1_60
Rect_60
GLCM2_60
Dens_60
Assym_60
NDVI_60
BordLngth_60
GLCM3_60
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

Coarseness Level 80




Coarseness Level 100
BrdIndx_100
Area_100
Round_100
Bright_100
Compact_100
ShpIndx_100
Mean_G_100
Mean_R_100
Mean_NIR_100
SD_G_100
SD_R_100
SD_NIR_100
LW_100
GLCM1_100
Rect_100
GLCM2_100
Dens_100
Assym_100
NDVI_100
BordLngth_100
GLCM3_100
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1


Coarseness Level 120
BrdIndx_120
Area_120
Round_120
Bright_120
Compact_120
ShpIndx_120
Mean_G_120
Mean_R_120
Mean_NIR_120
SD_G_120
SD_R_120
SD_NIR_120
LW_120
GLCM1_120
Rect_120
GLCM2_120
Dens_120
Assym_120
NDVI_120
BordLngth_120
GLCM3_120
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1



Analysis: 1. brdindx and shpindx are highly correlated across all coarseness levels 2. bright is highly correlated with mean_G, mean_R, and mean_NIR 3. SD_G, SD_R, SD_NIR are highly correlated among each other 4. rect is inversely correlated with shpindx, compact, round, and brdindx

Conclusions

1. classification can be attempted with only one coarsensss level, or levels 20 and 140
2. Brdindx, Shpindx, compact, and round can be omitted from the model since rect is highly correlated with all of them
3. Mean_G, Mean_R, and Mean_NIR can be omitted from the model since bright is highly correlated with all of them
4. SD_G, SD_R, or SD_NIR should be used in the model; whichever produces the best results