Danmarks Tekniske Universitet

Introduction to Machine Learning and Data mining

Assignment 01 Data Visualization

Assignment 01

By:
Hsuan Liang, s166126
Martin Hatting Petersen, s144234

Lecturer:
Morten Mørup

28-02-2017

Technical University of Denmark

Contents

1 Description of data set

The data set was used in the CoIL 2000 Challenge [1] with real data on customers of an insurance company. The attributes contain sociodemographic data derived from zip codes and product usage data of the customers. Since the sociodemographic data is collected by area, each sociodemographic attribute (6 to 43) reflects the percentage of people having the characteristics in the same area. Many customers share the same attributes of this type indicates that they live in the same area. Product usage data is specific to each customer.

This data set can be found at [2]. It has not been transformed but separated into two set as training set and test set, each containing 5822 and 4000 observations. As previously mentioned, the data set was used in a data mining contest and a few researches were done on the results of the contest. Originally, the last attribute was selected as the target variable and the task of the contest was to determine candidate customers of having the caravan insurance policy and give reasons to the result. Fig. 1 shows the result of the contest summarized in [3].

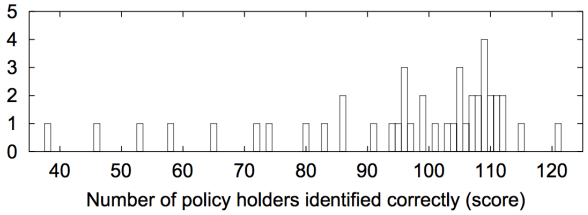

Figure 1: Distribution of scores achieved by 43 entries in CoIL 2000 Challenge.

The article also discuss several machine learning techniques adopted by different entries and analyzed the performance of each. The author drew conclusions on not only the methods applied in the contest but also the importance of giving attention to the issue of statistical significance.

The primary aim is to classify the last attribute, attribute 86, as 0 or 1, which indicates the whether a customer have the caravan insurance policy. However, classification on other attributes, e.g., customer type, number of cars etc., could also be done. Similar problems can be solved by logistic regression. Another possible approach to this data set would be carrying out association mining on the attributes. Since many pairs of attributes with high correlation are derived from the same entities, e.g., the amount of contribution one spend on a certain type of product correlates to the number of the product one have of that type (as shown in table: 2), we can discover if there exists other underlying associations.

2 Explanation of the attributes

We have chosen to only describe some of the attributes of the data since it contains 86 attributes. So we choose some different attributes and describe these.

1. Customer subtype is a Discrete and Nominal attribute.
2. Number of houses is a Discrete and Ratio attribute.
3. Average size of household is Discrete and an Interval attribute.
4. Average age is Discrete and a Interval attribute.
5. Marital status is a Discrete and Nominal attribute.
6. Educational level is a Discrete and Interval attribute.
7. Average income class is a Discrete and Interval attribute.
8. Contribution policies is a Discrete and Interval attribute.
9. Contribution insurances is a Discrete and Interval attribute.
10. Number of insurances for different accidents is a Discrete and Ratio attribute.
11. Number of policies for a given item is a Discrete and Ratio attribute.

All the attributes are discrete since they only contain specific values. Many of them are intervals since they are sociodemographic data describing the classes of the people with the same zip codes. Some are ratios since zero means the absence of the measured unit whereas others are nominal since they are some specific classes that have no order or any other way of ranking them. A few are ordinal which you cannot take the difference between the classes since that would make no sense.
The above is just a summary of the key attributes and the remaining have the same characteristics. Our data set has no missing values or corrupted data, so we won't cover this. Below we have conducted some summary statistics on the attributes.

Table 1: Summary statistics.

Attributes Statistics	Type	Mean	Std	Min	Max	Mode
Customer subtype	Nominal	24.253	12.847	1.000	41.000	33.000
Number of houses	Ratio	1.111	0.406	1.000	10.000	1.000
Average size of households	Interval	2.679	0.790	1.000	5.000	3.000
Average age	Interval	2.991	0.815	1.000	6.000	3.000
Marital status	Nominal	6.183	1.909	0.000	9.000	7.000
Educational level	Interval	4.572	2.298	0.000	9.000	5.000
Average income class	Interval	3.784	1.318	0.000	9.000	3.000
Contribution policies	Interval	2.970	2.921	0.000	8.000	0.000
Contribution insurances	Interval	1.828	1.879	0.000	8.000	0.000
Number of insurances	Ratio	0.570	0.562	0.000	7.000	1.000

3 Data visualization and PCA analysis

To investigate if there are any outliers in the data set we have made a boxplot of the chosen attributes. These plots can be seen below:

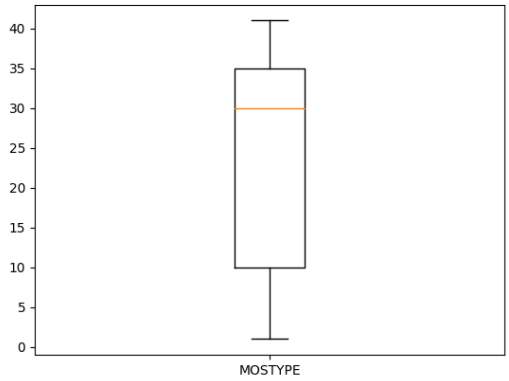

Figure 2: Boxplot of the customer subtype.

We divided the attributes into two boxplots since the first attribute has a larger range and the rest is much smaller so we plot those below:

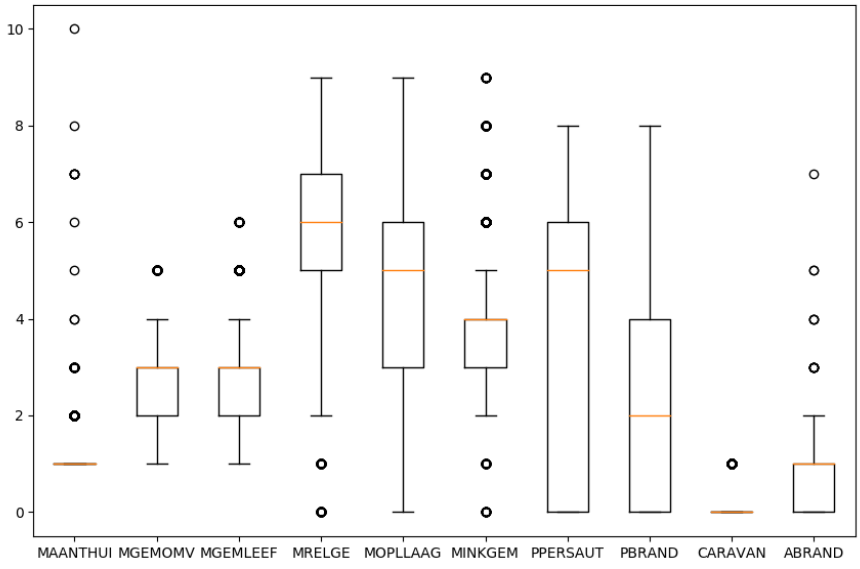

Figure 3: Boxplot of the remaining attributes.

We see that the first box has a lot of outliers and that might be due to that this attribute has a lot of 1's. This means that almost any other value than 1 would be classified as an outlier. But we don't consider this an outlier since they are plausible. Also, CARAVAN has an outlier of value 1 but since we don't have a lot of data with CARAVAN value 1 but many 0's, it influences how the boxplot classifies an outlier. We also looked at how many customers has CARAVAN 1 and it turned out to be around 300, which is relatively few compared to 5822 observed customers in total. We investigated if the attributes are normal distributed and not many of them are even close but the variables closest to being normal distributed is: MGEMOMV (Average size of household), MGEMLEEF (Average age), MOPLLAAG (Lower level education), MINKGEM (Average income). These are plotted below:

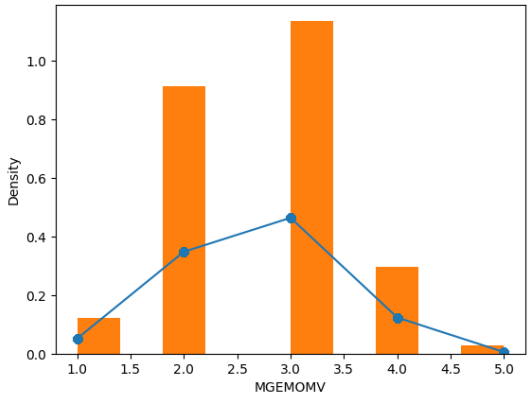

Figure 4: Histogram of the Average size of household.

This is not very close to a normal distribution but much closer than many of the other attributes.

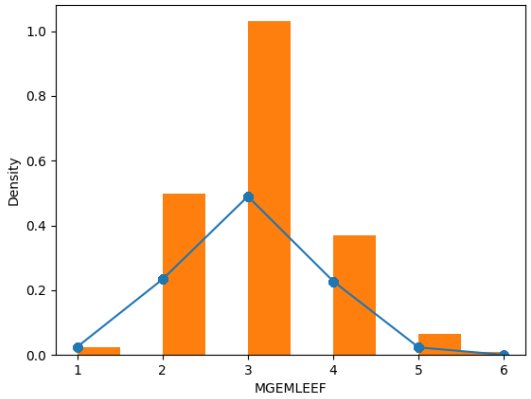

Figure 5: Histogram of the Average age.

This is too not to close to a normal distribution but like the first one this might be due to that these attributes are discrete with not much spread.

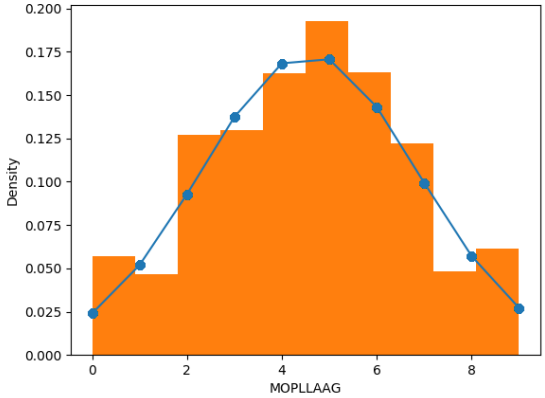

Figure 6: Histogram of the Lower level education.

This is very close to a normal distribution and this might be because the attribute has a larger spread.

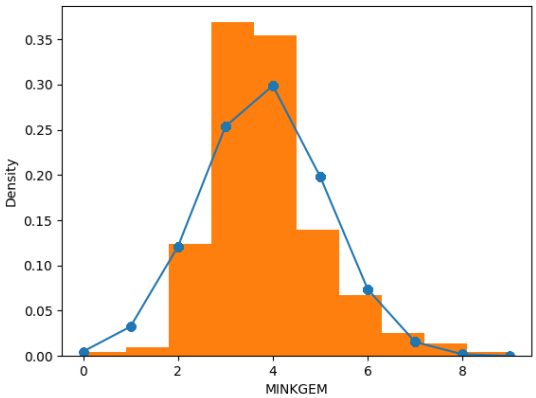

Figure 7: Histogram of the Average income.

The rest of the attributes are not close to having a normal distribution but the attribute MOPLLAAG is the closest to normal distribution of all attributes. Next, we will look at the attributes and check if any of them are closely correlated. We only look at the attributes which have a higher correlation than 0.9. And when we look closer at these we see that they make perfectly sense to be correlated. The ones with the highest correlations are:

Table 2: A table of the attributes which has a higher correlation than 0.9

Attribute name 1	Attribute name 2	Cor
Customer subtype	Customer main-type	0.9927
Cont. 3. part. insurance (agri.)	Num. of 3. part. insurance (agri.)	0.9876
Cont. priv. insurance	Num. of priv. insurance	0.9814
Cont. family acc. ins. pol.	Num. of family acc. ins. pol.	0.9800
Cont. moped pol.	Num. of moped pol.	0.9697
Cont. social security ins. pol.	Num. of social security ins. pol.	0.9662
Cont. trailer pol.	Num. trailer pol.	0.9661
Cont. lorry pol.	Num. of lorry pol.	0.9487
Cont. disab. ins. pol.	Num. of disab. ins. pol.	0.9484
Cont. bicycle pol.	Num. of bicycle pol.	0.9359
Cont. tractor pol.	Num. tractor pol.	0.9298
Cont. car policies	Num. of car policies	0.9162
Cont. agricultural mach. pol.	Num. of agricultural mach. pol.	0.9097
Cont. motorcycle pol.	Num. of motorcycle pol.	0.9049
Cont. surfboard pol.	Num. of surfboard pol.	0.9044
Cont. del. van pol.	Num. of del. van pol.	0.9030

Since our data seems correlated with some of the attributes we would exclude one of the attributes from each pair. Then it would be possible to do a classification task but not with logistic regression since our data set is not linearly separable; instead, we would use support-vector machines (SVM) to solve this problem since we don't have enough data to train a deep-neural-network (DNN).

To further visualize the data we perform a PCA on the data to better visualize. First we perform a variation analysis on the PCA in the following plot:

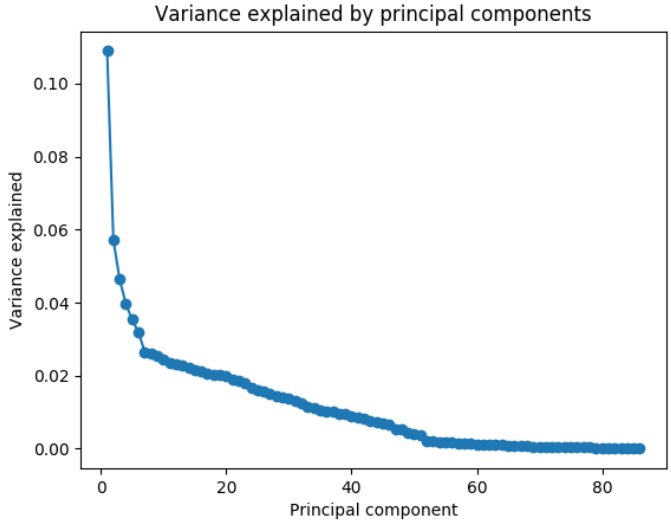

Figure 8: Variation explained as a function of number of principal components.

So we choose 10 principle directions since that explain a lot of the variance on the data. Below we plot the principal directions for the first 4 principal components:

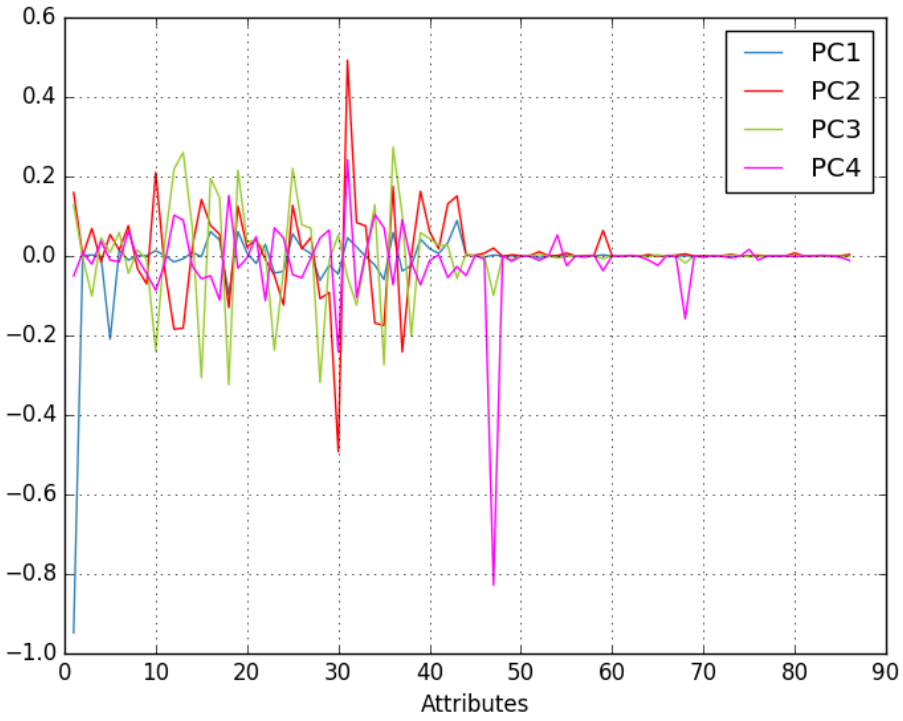

Figure 9: Direction of the first four principle components.

Now, we summarize the directions of the dominating principle components by plotting the first 4 principle components in the above diagram as they show prominent peaks. The first two peaks of PC1 at attributes 1 and 5 are MOSTYPE (Customer subtype) and MOSHOOFD (Customer main type) respectively. Two peaks of PC2 at attributes 30 and 31 are MHHUUR (Rented house) and MHKOOP (Home owners) respectively. PC3, however, has multiple peaks at similar heights. Finally, the peak of PC4 at attribute 47 corresponds to PPERSAUT (Contribution car policies). We could easily see that the two principle directions of PC1 and PC2 respectively are pairwise correlated, which justifies removing one attribute from the highly correlated pairs. Also observe that the directions of the four principle components (PC4 has one peak at 47) at attributes 44 to 86 converges to zero. These attributes describe the product usage data (non-sociodemographic data) that varies between each customer.

So from this point we choose 10 principal components since they provide a good estimate of the variance and shrinks the attribute number quite a bit. Then we make a plot of the data projected onto the first 10 principal components. This is plotted with a special plot called "parallel coordinates" as shown below:

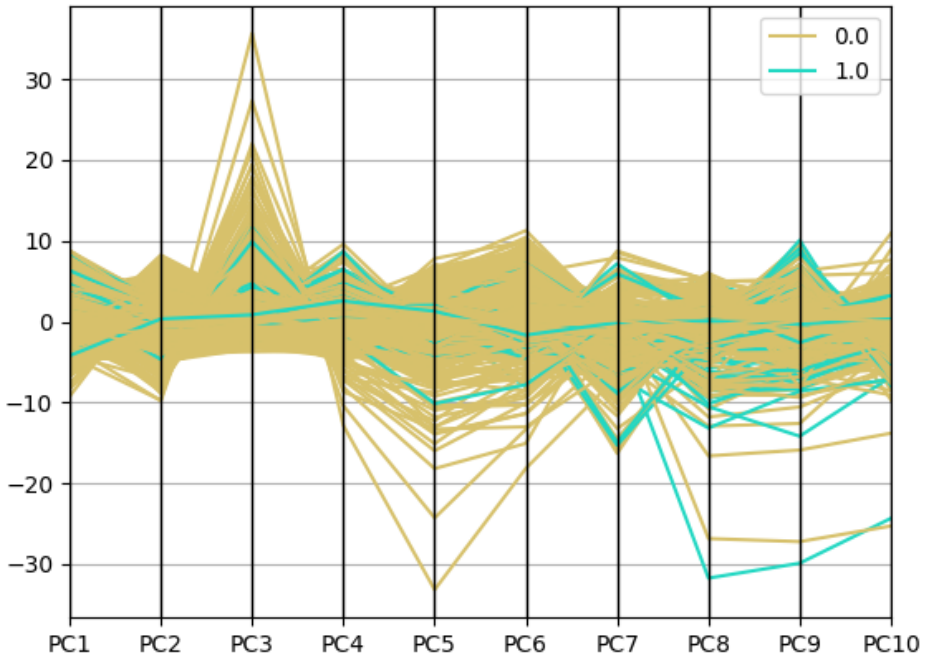

Figure 10: Data projected onto the first 10 principal components.

We see that the principal components have a hard time distinguishing between the two classes. Class 0 is where CARAVAN = 0 and for class 1 CARAVAN = 1. So we might have to use a more advanced approach to distinguish between the two classes.

4 Conclusion

In section 3, PCA does capture the variance in customer type and house condition as shown in the figure (9). However, numerous pairs of correlating attributes indicates that we could remove the one of the two attributes with close relation and carry out analysis again on lesser dimensions. Since the sociodemographic data are mostly interval attributes and the customer type is nominal, it is also worth transforming the values into one-out-of-k coding. As shown in Figure 10, PCA alone is insufficient to divide the classes as seen in figure (10), yet, the number of observations and attributes should allow us to carry out classification by other means.

5 Appendix

```
import sys
path_to_lib = '/home/martin/Dropbox/DTU/6. Semester/02450 - Machine Learning/
    Toolbox/02450Toolbox_Python/Scripts'
sys.path.insert(0, path_to_lib)

from matplotlib.pyplot import (figure, subplot, plot, xlabel, ylabel, title,
yticks, show,legend,imshow, cm, hold, boxplot,xticks,hist)
from scipy.linalg import svd
import scipy.stats as stats

import numpy as np
from pandas.tools.plotting import parallel_coordinates
import pandas as pd


# Load the data
X = np.loadtxt('./insuranceCompany_Data/ticdata2000.txt')

N = X.shape[0]
M = X.shape[1]

chosenAttributes = np.array([0,1,2,3,9,17,41,46,58,85,79])
chosenAttributesNames = np.array(['MOSTYPE','MAANTHUI','MGEMOMV','MGEMLEEF','
    MRELGE','MOPLLAAG','MINKGEM','PPERSAUT','PBRAND','CARAVAN','ABRAND'])


# boxplot of the data
figure()
boxplot(X[:,chosenAttributes[0]])
xticks([1], [chosenAttributesNames[0]])
figure()
boxplot(X[:,chosenAttributes[1:]])
xticks(range(1,len(chosenAttributes[1:])+1), chosenAttributesNames[1:])

for i in np.array([2,3,5,6]):
  Xs = sorted(X[:,chosenAttributes[i]])
  fit = stats.norm.pdf(Xs, np.mean(Xs), np.std(Xs))
  figure()
  plot(Xs,fit,'-o')
  hist(Xs,normed=True)
  xlabel(chosenAttributesNames[i])
  ylabel('Density')

# Subtract mean value from data
Xc = X - np.ones((N,1))*X.mean(0)
Xc = np.divide(Xc,Xc.std(0))

# PCA by computing SVD of Xc
U,S,VT = svd(Xc,full_matrices=False)

# Compute variance explained by principal components
rho = (S*S) / (S*S).sum()

# Plot variance explained
figure()
plot(range(1,len(rho)+1),rho,'o-')
title('Variance explained by principal components');
xlabel('Principal component');
ylabel('Variance explained');
```

```
# Selected number of principal components
k = 10

V = VT.T

# Project the data onto the considered principal components
Z = Xc @ V[:,:k]

# Plot PCA of the data
plt_data = pd.DataFrame(Z,columns=['PC1','PC2','PC3','PC4','PC5','PC6','PC7','PC8
    ','PC9','PC10'])
plt_data['res'] = np.round(X[:,85])

figure()
parallel_coordinates(plt_data, 'res')

# Find variance-covariance matrix and correlation
covVarMatrix = np.cov(X.T)
corMatrix = np.corrcoef(X.T) - np.diag(np.ones(M))

corAtt = np.where(corMatrix>0.9) # these are the attribute indexes which have a
    correlation above 0.9

# Show the plots
show()
```

```
import numpy as np
import pandas as pd

class InsuranceCompanyData:
    def __init__(self, data):
        self.data = data
        self.classGroup = self.readClass()
        self.L = self.readL()
    def readClass(self):
        classes = []
        classGroup = []
        with open("insuranceCompany_Data/class") as f:
            content = f.readlines()
            for line in content:
                if(line.isspace()):
                    classGroup.append(classes)
                    classes = []
                    continue
                split = line.split(" ")
                classes.append((int(split[0]), split[1], " ".join(split[2:]).
    strip("\n")))
            # append last group
            classGroup.append(classes)
        return classGroup
    def readL(self):
        def s(num):
            L = []
            with open("insuranceCompany_Data/L"+str(num)) as l:
                content = l.readlines()
                for line in content:
                    split = line.split(" ")
                    L.append((int(split[0]), " ".join(split[1:]).strip("\n")))
            return L
        Ls = []
        for i in range(5):
            Ls.append(s(i))
        return Ls
    def classGroupL(self, classGroup):
        L = self.L
        groupNum = classGroup[0][0]
        if groupNum == 1:
            l = L[0]
        elif groupNum == 4:
            l = L[1]
        elif groupNum == 5:
            l = L[2]
        elif 6 <= groupNum and groupNum <= 43:
            l = L[3]
        elif 44 <= groupNum and groupNum <= 64:
            l = L[4]
        else:
            print("Numerical")
            return
        for i in l:
            print(str(i[0])+" "+i[1])
    def classGroupStat(self, classGroup, code=False):
        data = self.data
        classDict = {}
        for classes in classGroup:
            colName = classes[2] if not code else " ".join([classes[1],classes
    [2]])
```

```
            classDict[colName] = data[:, classes[0]-1]
        return pd.DataFrame(classDict).describe()
    def stat(self, groupNum, showMinMax=False):
        group = self.classGroup[groupNum]
        attr = (group[0][0]-1, group[0][0]+len(group)-1)
        self.classGroupL(group)
        # print(self.data[0, attr[0]:attr[1]])
        df = self.classGroupStat(group)
        df = df.truncate() if showMinMax else df.truncate(after='std')
        print(df)
        colName = df.columns.values
        maxMeanInGroup, maxStdInGroup = np.argmax(df.values[1]), np.argmax(df.
    values[2])
        minMeanInGroup, minStdInGroup = np.argmin(df.values[1]), np.argmin(df.
    values[2])
        print(maxMeanInGroup, maxStdInGroup, minMeanInGroup, minStdInGroup)
        print("Max mean:\t", colName[maxMeanInGroup][:29], "\t\t", df.values[1][
    maxMeanInGroup])
        print("Min mean:\t", colName[minMeanInGroup][:29], "\t\t", df.values[1][
    minMeanInGroup])
        print("Max std:\t", colName[maxStdInGroup][:29], "\t\t", df.values[2][
    maxStdInGroup])
        print("Min std:\t", colName[minStdInGroup][:29], "\t\t", df.values[2][
    minStdInGroup])
    def summary(self, start=0, end=86):
        mean = []
        std = []
        for i in self.classGroup:
            df = self.classGroupStat(i)
            mean.append(j for j in df.values[1])
            std.append(df.values[2])
        mean, std = [i for j in mean for i in j], [i for j in std for i in j]
        mean, std= mean[start:end], std[start:end]
        print("mean over each attribute:\n", "max\t", np.argmax(mean, 0), np.amax
    (mean), "\n min\t", np.argmin(mean)+1, np.amin(mean))
        print("std over each attribute:\n", "max\t", np.argmax(std), np.amax(std)
    , "\n min\t", np.argmin(std)+1, np.amin(std))
```

```
from readClass import *
import numpy as np
import pandas as pd

data = np.loadtxt('./insuranceCompany_Data/ticdata2000.txt')

X = InsuranceCompanyData(data)
print("InsuranceCompanyData\n")
print("# of observations:\t", len(X.data))
print("# of classes:\t\t", len(X.data[0]))
print("# of groups:\t\t", len(X.classGroup))
print()
[0, 1, 2, 3, 8, 10]
# df = X.stat(8).values
# print(df[1][0])
for i in range(len(X.classGroup)):
    X.stat(i, True)
X.summary(1, 4)

# Y = X.data[12, 5:42]
# for i in X.data:
# counts = []
# inc = 0
# df = X.data
# for i in range(len(df)):
#     print(i)
#     count = 0
#     dlist = []
#     Y = df[i, 5:42]
#     for j in range(len(df)):
#         if (df[j, 5:42] == Y).all():
#             count+=1
#             dlist.append(j)
#     for d in dlist:
#         np.delete(df, d, 0)
#     counts.append(count)
# print(counts)
```

References

[1] P. van der Putten and M. van Someren (eds). Coil challenge 2000: The insurance company case. June 2000. Also a Leiden Institute of Advanced Computer Science Technical Report 2000-09.

[2] The insurance company benchmark (coil 2000). `https://archive.ics.uci.edu/ml/machine-learning-databases/tic-mld/tic.html`.

[3] Charles Elkan. Magical thinking in data mining: Lessons from coil challenge 2000. `http://cseweb.ucsd.edu/~elkan/kddcoil.pdf`.