Universidade Federal de Uberlândia

PGC308A Tópicos Especiais em Sistemas de Computação 2: Internet do Futuro

Rodrigo Moreira – moreira_r@outlook.com

2017-2

1 Task IV: Reduce the Number of Device Statistics to Estimate the Service Metric

1. Construct a training set and a test set from the trace as above.

   Done.

2. Build all subsets of the feature set X that contain either one or two features. Compute the models for each of these sets for linear regression over the training set. Plot a histogram of the error values (NMAE) of all the models for the test set. Identify the feature set that produces the model with the smallest error and give the device statistic(s) in this set.

   The figure below depicts the histogram for NMAE of the feature combination. The model that produces a smallest error is ['file.nr','ldavg.1] with 0,1088 or ≈ 10.88%:

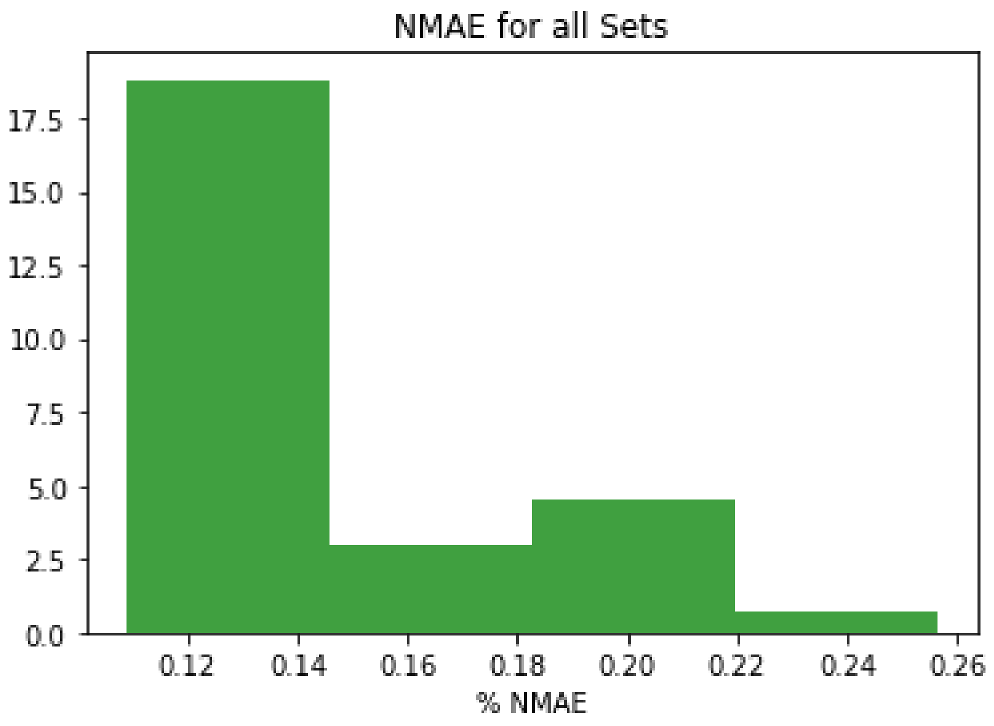
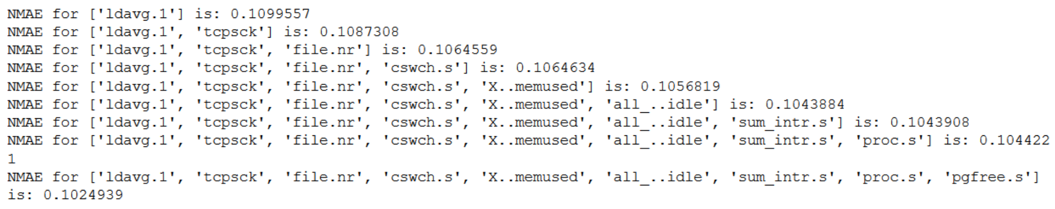


Figure 1: Histogram

3. Method 2: Linear univariate feature selection. Compute the model for each of these nine sets for linear regression over the training set and compute the error (NMAE) of these models over the test set. Produce a plot that shows the error value in function of the set k.



```
NMAE for ['ldavg.1'] is: 0.1099557
NMAE for ['ldavg.1', 'tcpsck'] is: 0.1087308
NMAE for ['ldavg.1', 'tcpsck', 'file.nr'] is: 0.1064559
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s'] is: 0.1064634
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused'] is: 0.1056819
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle'] is: 0.1043884
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle', 'sum_intr.s'] is: 0.1043908
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle', 'sum_intr.s', 'proc.s'] is: 0.104422
1
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle', 'sum_intr.s', 'proc.s', 'pgfree.s']
is: 0.1024939
```

Figure 2: NMAE Series

The Figure 3 illustrates a decrease in the error metric as more features are added:

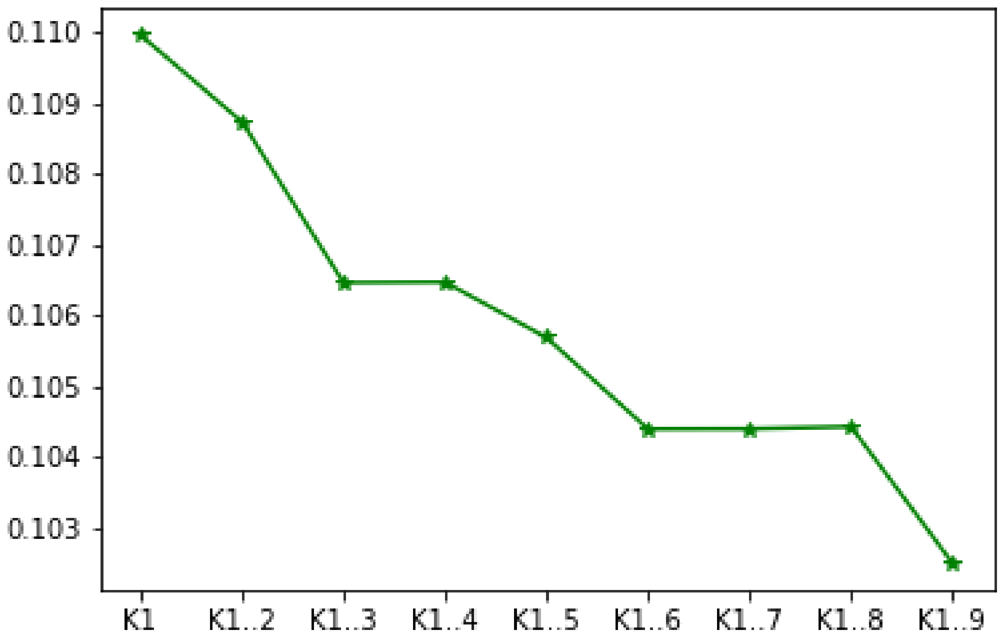


Figure 3: Plot – NNMAE Series

4. Describe your observations and conclusions.

   With the execution of the experiments it is possible to note that there are characteristics that in the perspective of linear regression, they have great representativity in the coefficient and finally in the predicion. That is, certain characteristics are dominant and more influential than others. Although this occurs in general, the joining of all characteristics decreases the error rate in the forecast. Another important aspect to note is a problem when we have a dataset with several features. The compuitonal resources needed to ensure that learning mechanism runs properly are supposed to be heavily used. So, to deal with the problem of several features and a large computational consume, we can choose the better features, that are more influential than others.

The codes used to solve these questions are available in the following link: https://github.com/romoreira/MLN/blob/master/Main_Task_IV.py

Main_Task_IV

November 27, 2017

```
In [2]: from __future__ import division
        import numpy as np
        import matplotlib.pyplot as plt
        from sklearn import linear_model
        from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import confusion_matrix
        from sklearn import metrics
        from sklearn.utils.validation import column_or_1d
        import pandas as pd
        #from sklearn.cross_validation import train_test_split
        from sklearn.feature_selection import RFE
        from sklearn.feature_selection import SelectKBest
        from sklearn.feature_selection import chi2
        from scipy.stats import pearsonr
        import math
        import cmath

        def fit_interleaves_features(x_train, y_train, x_test):
            regr = linear_model.LinearRegression()
            regr.fit(x_train, y_train)
            y_pred = regr.predict(x_test)
            return y_pred

        def nmae(y_real, y_predito):

            #print("Antes")
            #print(y_real)
            #print(y_predito)

            #Set of the DataFrame configs
            y_real = pd.DataFrame(y_real)
            y_predito = pd.DataFrame(y_predito)

            #print("Depois")
            #print(y_real.iloc[:, y_real.columns != "TimeStamp"])
            #print("Media: %.2f" % y_real.iloc[:, y_real.columns !=
        "TimeStamp"].astype(float).mean())
            #print(y_predito)
```

```
    #print("Y real: ")
    #print(y_real.iloc[0][1])

    #Loop variables initializing
    somatorio = 0.0
    m = 0
    media = 0
    nmae_resultado = 0
    for m in range(len(y_real)):
        somatorio += abs((y_real.iloc[m][1] - y_predito.iloc[m]))
        m += 1


    #Ajustes
    media = y_real.iloc[:, y_real.columns != "TimeStamp"].astype(float).mean()
    media = media[0]
    somatorio = somatorio[0]


    #N. M. A. E. Accuracy Measures
    nmae_resultado = (somatorio/m)/media

    return nmae_resultado

def get_dataframe():
    csv_x = pd.read_csv('./data/X.csv', sep=',', header=None)
    csv_y = pd.read_csv('./data/Y.csv', sep=',', header=None)

    # Esse trecho de codigo retira a primeira linha do DataFrame (que contem os
nomes das colunas), cria uma novo DataFrame sem essa primeira linha,
    # depois adiciona as colunas na forma de indices
    new_header = csv_x.iloc[0]
    csv_x = csv_x[1:]
    csv_x.columns = new_header

    new_header = csv_y.iloc[0]
    csv_y = csv_y[1:]
    csv_y.columns = new_header

    csv_x['TimeStamp'] = pd.to_numeric(csv_x['TimeStamp'])
    csv_x['all_..idle'] = pd.to_numeric(csv_x['all_..idle'])
    csv_x['X..memused'] = pd.to_numeric(csv_x['X..memused'])
    csv_x['proc.s'] = pd.to_numeric(csv_x['proc.s'])
    csv_x['cswch.s'] = pd.to_numeric(csv_x['cswch.s'])
    csv_x['file.nr'] = pd.to_numeric(csv_x['file.nr'])
    csv_x['sum_intr.s'] = pd.to_numeric(csv_x['sum_intr.s'])
    csv_x['tcpsck'] = pd.to_numeric(csv_x['tcpsck'])
    csv_x['pgfree.s'] = pd.to_numeric(csv_x['pgfree.s'])

    csv_y['TimeStamp'] = pd.to_numeric(csv_y['TimeStamp'])
    csv_y['DispFrames'] = pd.to_numeric(csv_y['DispFrames'])

    return csv_x, csv_y


def dataset_headers(dataset):
    # Monto uma lista com os nomes das colunas
    return list(dataset.columns.values)


def binarize_y(y):
    # Adiciona a Y (Target) valores binarios para o SLA Conformance
    i = 0
    sla_conformance_y = np.array([])

    for i in range(len(y)):
        if y.iloc[i]['DispFrames'] >= 18:
            sla_conformance_y = np.append(sla_conformance_y, 1.0)
        else:
            sla_conformance_y = np.append(sla_conformance_y, 0.0)
        i += 1

    return sla_conformance_y


# ---------Task III ----------------------------
#_______________________________________________


csv_x, csv_y = get_dataframe()
x_train, x_test, y_train, y_test = train_test_split(csv_x, csv_y,
test_size=0.30)

#---------Question 2----------------------------
#_______________________________________________


#y_train = column_or_1d(y_train, warn=False)
y_train = column_or_1d(y_train.iloc[:, y_train.columns != "TimeStamp"],
warn=False)
model = linear_model.LinearRegression()
rfe = RFE(model, 3)
fit = rfe.fit(x_train.iloc[:, x_train.columns != "TimeStamp"],y_train)
n_features = fit.n_features_
selected_feature = fit.support_
```

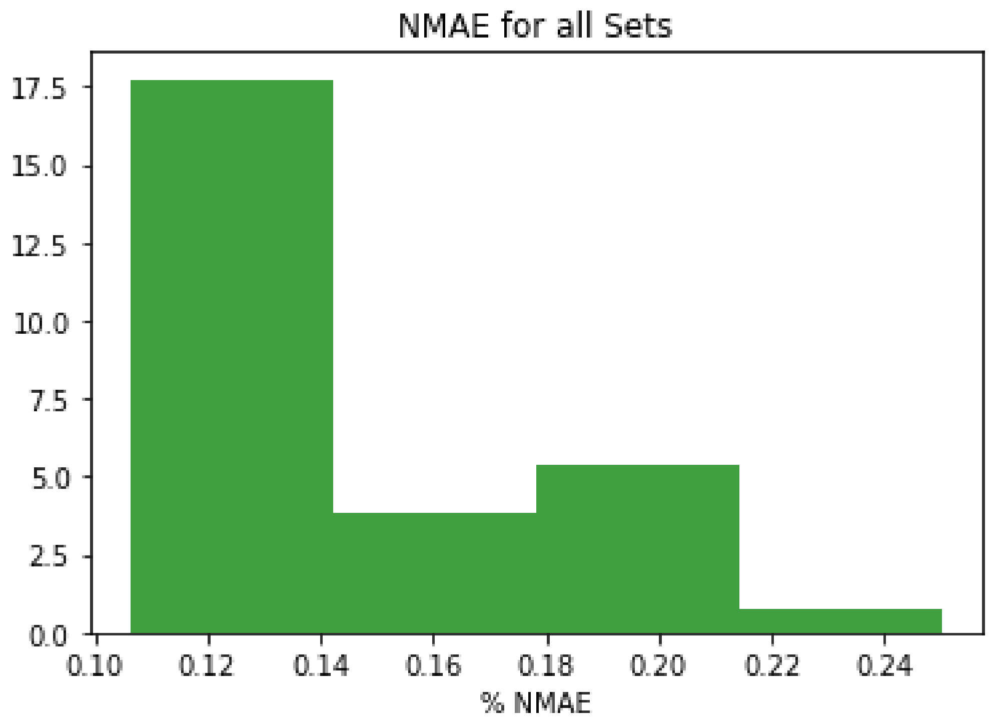
```
feature_ranking = fit.ranking_
print("Num Features: %.f" % n_features.astype(float))
print("Selected Features: %s" %selected_feature)
print("Feature Ranking: %s"  %feature_ranking)

#Teste para conjunto com 1 caracteristica
x_train_bkp = x_train
y_train_bkp = y_train
x_test_bkp = x_test
y_test_bkp = y_test
for column in x_train:
    if(column == "TimeStamp"):
        print()
    else:
        x_train = x_train.iloc[:,x_train.columns != "TimeStamp"][column]
        x_train = pd.DataFrame(x_train, columns = [column])

        x_test = x_test.iloc[:,x_test.columns != "TimeStamp"][column]
        x_test = pd.DataFrame(x_test, columns = [column])


        print("NMAE for ['%s'] is: %.3f" % (column, nmae(y_test,
fit_interleaves_features(x_train, y_train, x_test))))
        #print(fit_interleaves_features(x_train, y_train.iloc[:,y_train.columns
!= "TimeStamp"], x_test))
        x_train = x_train_bkp
        x_test = x_test_bkp

##Teste para conjunto com 2 caracteristicas
x_train = x_train_bkp
x_test = x_test_bkp
features = list(x_train.columns)

#Crio um numpy array para armazenar as NMAE para posteriormente plotar o
Histogram
nmae_array = np.array([])

i = 1
while i < len(features):
    j = i+1
    while j < len(features):

        x_train = x_train.iloc[:, [i,j]]

        x_test = x_test.iloc[:, [i,j]]

        nmae_array = np.append(nmae_array, nmae(y_test,
fit_interleaves_features(x_train, y_train, x_test)))
```

```
        ##print(nmae_array)

        print("NMAE for ['%s','%s] is: %.4f" % (features[i], features[j],
nmae(y_test, fit_interleaves_features(x_train, y_train, x_test))))
        j += 1
        x_train = x_train_bkp
        x_test = x_test_bkp
    i += 1

#NAMAE Histogram
plt.hist(nmae_array, 4, normed=1, facecolor='green', alpha=0.75)
plt.xlabel('% NMAE')
plt.title("NMAE for all Sets")
plt.show()


# ---------Task III ---------------------

csv_x, csv_y = get_dataframe()

x_train, x_test, y_train, y_test = train_test_split(csv_x, csv_y,
test_size=0.30)

#_______________________________________________
#---------Question 3----------------------------
#_______________________________________________

features= np.array([])
for column in x_train:
    features= np.append(features,column)

correlation_array = np.float32([])
i = 1
while i < len(features):
    x_column = np.array(x_train[features[i]])
    y_train = y_train.iloc[:, y_train.columns == 'DispFrames']
    y = np.array(y_train)
    y_values = column_or_1d(y, warn=False)

    #print(pearsonr(x_column.astype(float), y_values.astype(float)))
    #print(pearsonr(x_column.astype(float), y_values.astype(float))[0])
    correlation_array = np.append(correlation_array,
pearsonr(x_column.astype(float), y_values.astype(float))[0].astype(float))
    i += 1

temp = np.column_stack((features[1:],correlation_array))
correlation_set = pd.DataFrame(temp,columns=['Feature','Correlation'])
```

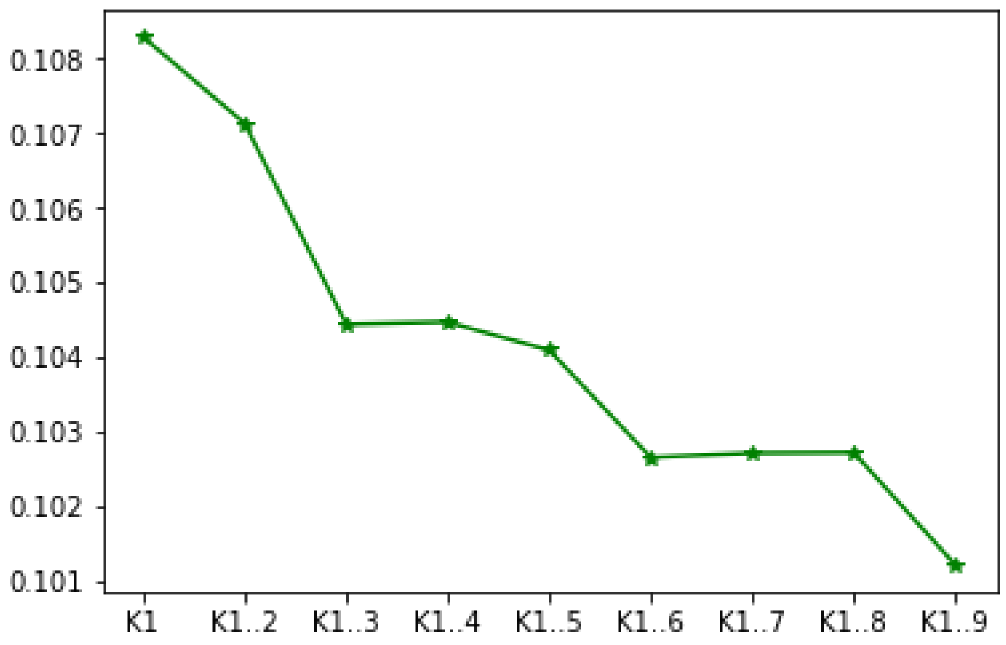
```
#print(correlation_set)

i = 0
correlation_square = np.float32([])
while i < len(correlation_set):
    correlation_square = np.append(correlation_square,
float(correlation_set.loc[i, 'Correlation'])*float(correlation_set.loc[i,
'Correlation']))
    i += 1

correlation_set['R2'] = pd.to_numeric(correlation_square)
#print(correlation_set)
correlation_set = correlation_set.sort_values(by='R2', ascending=False)
print(correlation_set)

features_list = list(correlation_set['Feature'])
#print(features_list)

x_train = x_train[features_list]
x_test = x_test[features_list]

x_train_bkp = x_train
y_train_bkp = y_train
x_test_bkp = x_test
y_test_bkp = y_test

#Crio um numpy array para armazenar as NMAE para posteriormente plotar o
Histogram
nmae_array = np.array([])

#print(x_train.iloc[:,0:1])

#print(features_list)

i = 1
while i <= len(features_list):

    x_train = x_train.iloc[:, 0:i]
    x_test = x_test.iloc[:, 0:i]

    nmae_array = np.append(nmae_array, nmae(y_test,
fit_interleaves_features(x_train, y_train, x_test)))

    ##print(nmae_array)

    print("NMAE for %s is: %.7f" % (features_list[:i], nmae(y_test,
fit_interleaves_features(x_train, y_train, x_test))))
    j += 1
```

```
            x_train = x_train_bkp
            x_test = x_test_bkp
            i += 1

        plt.plot(['K1','K1..2','K1..3','K1..4','K1..5','K1..6','K1..7','K1..8','K1..9'],
        nmae_array, '-g*')
        plt.show()
```

```
Num Features: 3
Selected Features: [False  True False False False False  True  True False]
Feature Ranking: [2 1 3 5 4 6 1 1 7]

NMAE for ['all_..idle'] is: 0.203
NMAE for ['X..memused'] is: 0.188
NMAE for ['proc.s'] is: 0.250
NMAE for ['cswch.s'] is: 0.145
NMAE for ['file.nr'] is: 0.128
NMAE for ['sum_intr.s'] is: 0.208
NMAE for ['ldavg.1'] is: 0.110
NMAE for ['tcpsck'] is: 0.125
NMAE for ['pgfree.s'] is: 0.261
NMAE for ['all_..idle','X..memused] is: 0.1715
NMAE for ['all_..idle','proc.s] is: 0.1999
NMAE for ['all_..idle','cswch.s] is: 0.1317
NMAE for ['all_..idle','file.nr] is: 0.1254
NMAE for ['all_..idle','sum_intr.s] is: 0.1847
NMAE for ['all_..idle','ldavg.1] is: 0.1086
NMAE for ['all_..idle','tcpsck] is: 0.1249
NMAE for ['all_..idle','pgfree.s] is: 0.2024
NMAE for ['X..memused','proc.s] is: 0.1860
NMAE for ['X..memused','cswch.s] is: 0.1414
NMAE for ['X..memused','file.nr] is: 0.1275
NMAE for ['X..memused','sum_intr.s] is: 0.1710
NMAE for ['X..memused','ldavg.1] is: 0.1100
NMAE for ['X..memused','tcpsck] is: 0.1245
NMAE for ['X..memused','pgfree.s] is: 0.1852
NMAE for ['proc.s','cswch.s] is: 0.1443
NMAE for ['proc.s','file.nr] is: 0.1275
NMAE for ['proc.s','sum_intr.s] is: 0.2037
NMAE for ['proc.s','ldavg.1] is: 0.1101
NMAE for ['proc.s','tcpsck] is: 0.1247
NMAE for ['proc.s','pgfree.s] is: 0.2504
NMAE for ['cswch.s','file.nr] is: 0.1168
NMAE for ['cswch.s','sum_intr.s] is: 0.1453
NMAE for ['cswch.s','ldavg.1] is: 0.1100
NMAE for ['cswch.s','tcpsck] is: 0.1199
NMAE for ['cswch.s','pgfree.s] is: 0.1448
NMAE for ['file.nr','sum_intr.s] is: 0.1246
NMAE for ['file.nr','ldavg.1] is: 0.1062
NMAE for ['file.nr','tcpsck] is: 0.1134
NMAE for ['file.nr','pgfree.s] is: 0.1273
NMAE for ['sum_intr.s','ldavg.1] is: 0.1100
NMAE for ['sum_intr.s','tcpsck] is: 0.1190
NMAE for ['sum_intr.s','pgfree.s] is: 0.2072
NMAE for ['ldavg.1','tcpsck] is: 0.1084
NMAE for ['ldavg.1','pgfree.s] is: 0.1097
NMAE for ['tcpsck','pgfree.s] is: 0.1245
```



```
       Feature            Correlation            R2
6      ldavg.1    -0.8452241700647415  7.144039e-01
7       tcpsck    -0.8057952356613306  6.493060e-01
4      file.nr    -0.7920578931575714  6.273557e-01
3      cswch.s    -0.7646755494205585  5.847287e-01
1   X..memused     0.5945357206214785  3.534727e-01
0   all_..idle     0.5707560376823818  3.257625e-01
5   sum_intr.s     0.5196470180608158  2.700330e-01
2       proc.s    -0.2397494150403977  5.747978e-02
8     pgfree.s  0.0007781569636773916  6.055283e-07
NMAE for ['ldavg.1'] is: 0.1082847
NMAE for ['ldavg.1', 'tcpsck'] is: 0.1071311
NMAE for ['ldavg.1', 'tcpsck', 'file.nr'] is: 0.1044413
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s'] is: 0.1044716
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused'] is: 0.1040967
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle']
is: 0.1026605
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle',
'sum_intr.s'] is: 0.1027179
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle',
'sum_intr.s', 'proc.s'] is: 0.1027251
NMAE for ['ldavg.1', 'tcpsck', 'file.nr', 'cswch.s', 'X..memused', 'all_..idle',
'sum_intr.s', 'proc.s', 'pgfree.s'] is: 0.1012201
```



0.108
0.107
0.106
0.105
0.104
0.103
0.102
0.101
K1
K1..2
K1..3
K1..4
K1..5
K1..6
K1..7
K1..8
K1..9