preprocessing

November 3, 2024

[123]:
```
# -*- coding: utf-8 -*-
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
# implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
```

0.0.1 Adam Candrák/Mária Matušisková - 50%/50%

1 Imports

[124]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.feature_selection import SelectKBest, f_classif
import matplotlib.pyplot as plt
from numpy import mean
from numpy import std
from collections import Counter
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler, PowerTransformer, MinMaxScaler
```

1.1 Global variables

[125]:
```
target_column = 'mwra'
test_size = 0.3
random_state = 42
```

[126]:
```
connections_file = "../data/Connections.csv"
processes_file = "../data/Processes.csv"

connections = pd.read_csv(connections_file, sep='\t')
processes = pd.read_csv(processes_file, sep='\t')
```

Change the names of the columns

[127]:
```
c_connections = connections.rename(columns={
    "c.katana": "facebook",
    "c.android.chrome": "chrome",
    "c.android.gm": "gmail",
    "c.dogalize": "dogalize",
    "c.android.youtube": "youtube",
    "c.updateassist": "updateassist",
    "c.UCMobile.intl": "UCMobile.intl",
    "c.raider": "raider",
    "c.android.vending": "vending",
    "c.UCMobile.x86": "UCMobile.x86",
})

p_processes = processes.rename(columns={
    "p.katana": "facebook",
    "p.android.chrome": "chrome",
    "p.android.gm": "gmail",
    "p.dogalize": "dogalize",
    "p.android.vending": "vending",
    "p.android.packageinstaller": "packageinstaller",
    "p.system": "system",
    "p.android.documentsui": "documentsui",
    "p.android.settings": "settings",
    "p.android.externalstorage": "externalstorage",
    "p.android.defcontainer": "defcontainer",
    "p.inputmethod.latin": "inputmethod.latin",
    "p.process.gapps": "gapps",
    "p.simulator": "simulator",
    "p.android.gms": "google mobile services (gms)",
    "p.google": "google",
    "p.olauncher": "olauncher",
    "p.browser.provider": "browser provider",
    "p.notifier": "notifier",
    "p.gms.persistent": "gms.persistent",
})
```

Change the type of timestamp to int64 of connections' dataset:

```
[128]: c_connections['ts'] = pd.to_datetime(c_connections['ts']).astype(np.int64)
```

Change the type of timestamp to int64 of processes dataset:

```
[129]: p_processes['ts'] = pd.to_datetime(p_processes['ts']).astype(np.int64)
```

1.1.1 Merge datasets connections and processes:

```
[130]: merged_dataset = pd.merge(c_connections, p_processes, on=['imei', 'ts', 'mwra'])
merged_dataset
```

```
[130]:                     ts                 imei  mwra  facebook_x  chrome_x  \
0      1525514400000000000  3590433799317662188   1.0    10.99774  11.05477
1      1525514460000000000  3590433799317662394   1.0    11.08234   9.64636
2      1525514520000000000  3590433799317661834   0.0    11.49582  12.27416
3      1525514580000000000  8630330696303481289   0.0    10.50935  11.41774
4      1525514640000000000  8630330696303481149   0.0    10.25989  14.46448
…                        …                    …     …          …         …
15445  1525987200000000000  3590433799317661925   1.0    11.23638  12.54494
15446  1526140320000000000   863033069630348776   0.0    11.71795  13.86245
15447  1526140320000000000   863033069630348776   0.0    11.71795  13.86245
15448  1526338140000000000  3590433799317661206   1.0    15.51197  16.53309
15449  1526338140000000000  3590433799317661206   1.0    15.51197  16.53309

        gmail_x  dogalize_x   youtube  updateassist  UCMobile.intl  …  \
0       6.03999    12.49767   8.59956      14.00953       52.54470  …
1       8.64167    12.60788   9.84197      38.27736       44.56009  …
2      11.59681    12.99258   9.74923      57.41411       36.83333  …
3      14.43350    12.91018  13.93857      31.57549       41.34296  …
4      14.02728     8.58832  13.04853      49.47100       38.86755  …
…           …           …         …             …              …  …
15445  11.26646     9.03636  12.76080      40.59272       30.60551  …
15446  12.07446    10.10313  13.96660      45.07102       52.21975  …
15447  12.07446    10.10313  13.96660      45.07102       52.21975  …
15448  15.75924    11.99555  14.99913      43.95935       35.75699  …
15449  15.75924    11.99555  14.99913      43.95935       35.75699  …

       dogalize_y     gapps  simulator  facebook_y  \
0        95.23250  99.55387   82.64951    55.62534
1        73.67809  55.93619   27.33158    68.28812
2        49.43847  92.96630   54.04233    25.01599
3        71.37356   8.34277   87.09809     5.21806
4        14.58892  27.72954   81.20459    22.42807
…               …         …          …           …
```

```
15445    87.48802   1.93764   55.19853    61.24749
15446    54.55465  34.82606   68.63334    98.60112
15447    54.55465  34.82606   68.63334    98.60112
15448    74.14262  50.12292   55.73990    82.29958
15449    74.14262  50.12292   55.73990    82.29958

       google mobile services (gms)    google  olauncher  browser provider  \
0                          43.73958  28.79282    8.22474          73.26391
1                          67.18486  19.40350   19.26265          58.69464
2                          57.15110  60.38043   16.88231          55.62452
3                          98.58641  97.22889   37.30215          68.75315
4                          25.06680  73.26831   43.72205          78.80356
…                               …         …        …             …
15445                      27.68095   7.42961   73.85143          98.09352
15446                      20.39117  17.48535   97.03999          59.31267
15447                      20.39117  17.48535   97.03999          59.31267
15448                      34.64708  85.38809   68.57332          50.53580
15449                      34.64708  85.38809   68.57332          50.53580

       notifier  gms.persistent
0      25.28004        86.66346
1      90.54099        33.10194
2      16.82005        81.58652
3      26.44336        79.98101
4      16.55350        75.03307
…          …             …
15445  39.51722        29.41750
15446  97.53013         7.77545
15447  97.53013         7.77545
15448  61.50573        81.83581
15449  61.50573        81.83581

[15450 rows x 33 columns]
```

1.1.2 Check for missing values and duplicities

[131]:
```
has_nan = merged_dataset.isnull().values.any()

if has_nan:
    print("The dataset has NaN values.")
    print(merged_dataset.isnull().values)
else:
    print("No NaN values found in the dataset.")
```

```
No NaN values found in the dataset.
```

```
[132]: has_duplicity = merged_dataset.duplicated().any()

       if has_duplicity:
           print("The dataset has duplicity values.")
           print(merged_dataset[merged_dataset.duplicated()])
           print("Number of duplicate rows:", merged_dataset.duplicated().sum())
       else:
           print("No duplicity values found in the dataset.")
```

```
The dataset has duplicity values.
                        ts                 imei  mwra  facebook_x  chrome_x  \
95     1525520040000000000  8630330696303482303   1.0    10.82916  11.29582
108    1525520760000000000   863033069630348065   1.0    14.71992  14.01692
175    1525524720000000000   359043379931766353   0.0     8.46728  13.41079
300    1525532160000000000  8630330696303481669   1.0     9.87679   8.52849
303    1525532280000000000   359043379931766924   1.0    10.22849  11.46160
…                      …                  …   …         …       …
15445  1525987200000000000  3590433799317661925   1.0    11.23638  12.54494
15446  1526140320000000000   863033069630348776   0.0    11.71795  13.86245
15447  1526140320000000000   863033069630348776   0.0    11.71795  13.86245
15448  1526338140000000000  3590433799317661206   1.0    15.51197  16.53309
15449  1526338140000000000  3590433799317661206   1.0    15.51197  16.53309

        gmail_x  dogalize_x   youtube  updateassist  UCMobile.intl  …  \
95     12.66003    10.16379  16.04734      39.11921       41.94816  …
108    14.46679    10.37614  15.28242      54.88030       54.02226  …
175    11.08708     6.18605   9.60942      52.78819       56.42743  …
300     9.57042    11.81096   7.54489       7.94035       61.92247  …
303     8.69223    10.45511  12.44206      46.64713       52.17634  …
…         …         …       …           …            …  …
15445  11.26646     9.03636  12.76080      40.59272       30.60551  …
15446  12.07446    10.10313  13.96660      45.07102       52.21975  …
15447  12.07446    10.10313  13.96660      45.07102       52.21975  …
15448  15.75924    11.99555  14.99913      43.95935       35.75699  …
15449  15.75924    11.99555  14.99913      43.95935       35.75699  …

       dogalize_y     gapps  simulator  facebook_y  \
95       97.52469  95.01733   56.82875    81.39681
108      87.62723  75.66986   13.90231    10.51647
175      30.91028   1.84816   36.52292     8.24216
300      75.93998  80.95502   86.96990     1.35263
303      61.25695  57.35034   36.31488    67.14928
…             …         …         …          …
15445    87.48802   1.93764   55.19853    61.24749
15446    54.55465  34.82606   68.63334    98.60112
15447    54.55465  34.82606   68.63334    98.60112
15448    74.14262  50.12292   55.73990    82.29958
15449    74.14262  50.12292   55.73990    82.29958
```

```
       google mobile services (gms)    google  olauncher  browser provider  \
95                         15.49079  33.46497   47.35665           9.96695
108                        36.08753  32.65558   13.72486          74.68378
175                        85.18883  85.80315   28.04099           5.37063
300                        33.50430   7.26696   23.43879          90.34312
303                        38.96647  56.28646   88.96592          11.03558
…                               …         …         …               …
15445                      27.68095   7.42961   73.85143          98.09352
15446                      20.39117  17.48535   97.03999          59.31267
15447                      20.39117  17.48535   97.03999          59.31267
15448                      34.64708  85.38809   68.57332          50.53580
15449                      34.64708  85.38809   68.57332          50.53580

       notifier  gms.persistent
95      9.45719        90.77472
108    47.71853        10.51180
175    64.96796        16.24571
300    43.77779        19.04622
303    36.52814        89.92218
…          …             …
15445  39.51722        29.41750
15446  97.53013         7.77545
15447  97.53013         7.77545
15448  61.50573        81.83581
15449  61.50573        81.83581

[537 rows x 33 columns]
Number of duplicate rows: 537
```

1.1.3 Drop values which are not helpful for further training:

[133]:
```python
merged_dataset.drop('ts', axis=1, inplace=True)
merged_dataset.drop('imei', axis=1, inplace=True)
```

1.1.4 Outlier deletion

[134]:
```python
# Source: https://www.kaggle.com/code/marcinrutecki/outlier-detection-methods

def StandardDevDetection(data, n, columns):

    outliers_inx = []
    lower = 0
    upper = 0

    for column in columns:
        # Calculate mean and standard derivation of each column
        data_mean, data_std = mean(data[column], axis=0), std(data[column],␣
↪axis=0)
        print('column=', column, 'len=', len(data), 'mean=', data_mean, 'std=',␣
↪data_std)

        # Divide it to the three outliers in the standard deviations:
        cut_off = data_std * 3
        lower, upper = data_mean - cut_off, data_mean + cut_off
        print('column=', column, 'cutoff=', cut_off, 'lower=', lower, 'upper=',␣
↪upper)

        # Filter the dataframe:
        outliers = data[(data[column] < lower) | (data[column] > upper)].index
        print('Identified outliers:', len(outliers))

        outliers_inx.extend(outliers)

    outliers_inx = Counter(outliers_inx)
    multiple_outliers = list( k for k, v in outliers_inx.items() if v > n )

    data_uppper = data[data[column] > upper]
    data_lower = data[data[column] < lower]
    print('Total number of outliers is:', data_uppper.shape[0] + data_lower.
↪shape[0])

    return multiple_outliers


columns = merged_dataset.columns
result = StandardDevDetection(merged_dataset, 1, columns)

new_dataset = merged_dataset.drop(result, axis = 0).reset_index(drop=True)
```

```
column= mwra len= 15450 mean= 0.6255663430420711 std= 0.48397633567669496
column= mwra cutoff= 1.4519290070300848 lower= -0.8263626639880136 upper=
2.077495350072156
Identified outliers: 0
column= facebook_x len= 15450 mean= 10.962643722330096 std= 2.6723458605226362
column= facebook_x cutoff= 8.017037581567909 lower= 2.9456061407621874 upper=
18.979681303898005
Identified outliers: 30
column= chrome_x len= 15450 mean= 11.605878082200647 std= 2.5788799704504304
column= chrome_x cutoff= 7.736639911351292 lower= 3.8692381708493553 upper=
19.342517993551937
Identified outliers: 20
column= gmail_x len= 15450 mean= 12.255588944983819 std= 2.5607912382789277
column= gmail_x cutoff= 7.682373714836784 lower= 4.573215230147035 upper=
19.937962659820602
Identified outliers: 47
column= dogalize_x len= 15450 mean= 10.453980822653723 std= 2.2933559796410883
column= dogalize_x cutoff= 6.880067938923265 lower= 3.5739128837304577 upper=
17.334048761576987
Identified outliers: 79
column= youtube len= 15450 mean= 12.250241922330098 std= 2.55734941631557
column= youtube cutoff= 7.67204824894671 lower= 4.578193673383388 upper=
19.922290171276806
Identified outliers: 14
column= updateassist len= 15450 mean= 45.98381515598705 std= 12.502608454363255
column= updateassist cutoff= 37.507825363089765 lower= 8.475989792897288 upper=
83.49164051907681
Identified outliers: 42
column= UCMobile.intl len= 15450 mean= 45.88107211326861 std= 13.050892012462258
column= UCMobile.intl cutoff= 39.152676037386776 lower= 6.728396075881832 upper=
85.03374815065538
Identified outliers: 24
column= raider len= 15450 mean= 49.188615063430426 std= 13.337020785059845
column= raider cutoff= 40.01106235517953 lower= 9.177552708250893 upper=
89.19967741860995
Identified outliers: 38
column= vending_x len= 15450 mean= 49.60707255469255 std= 28.92403819144008
column= vending_x cutoff= 86.77211457432024 lower= -37.16504201962769 upper=
136.3791871290128
Identified outliers: 0
column= UCMobile.x86 len= 15450 mean= 49.76764124854368 std= 28.696862484297487
column= UCMobile.x86 cutoff= 86.09058745289246 lower= -36.32294620434878 upper=
135.85822870143613
Identified outliers: 0
column= packageinstaller len= 15450 mean= 11.078107833009707 std=
2.814283167101723
column= packageinstaller cutoff= 8.442849501305169 lower= 2.635258331704538
upper= 19.520957334314875
Identified outliers: 22
column= system len= 15450 mean= 10.999383213592232 std= 2.5257279118949145
column= system cutoff= 7.577183735684743 lower= 3.4221994779074887 upper=
18.576566949276973
Identified outliers: 26
column= documentsui len= 15450 mean= 11.08572180064725 std= 2.5594279088864607
column= documentsui cutoff= 7.678283726659382 lower= 3.4074380739878674 upper=
18.76400552730663
Identified outliers: 23
column= chrome_y len= 15450 mean= 12.138836054368932 std= 2.5379252214438917
column= chrome_y cutoff= 7.6137756643316745 lower= 4.5250603900372575 upper=
19.752611718700607
Identified outliers: 22
column= settings len= 15450 mean= 13.40750721618123 std= 1.898547862388675
```

```
column= settings cutoff= 5.695643587166025 lower= 7.711863629015205 upper=
19.103150803347255
Identified outliers: 48
column= gmail_y len= 15450 mean= 12.784054436245954 std= 2.4946624687029955
column= gmail_y cutoff= 7.483987406108986 lower= 5.300067030136968 upper=
20.26804184235494
Identified outliers: 104
column= externalstorage len= 15450 mean= 11.600245935922329 std=
2.586746058468083
column= externalstorage cutoff= 7.760238175404249 lower= 3.8400077605180805
upper= 19.360484111326578
Identified outliers: 19
column= defcontainer len= 15450 mean= 50.635713223300975 std= 12.747023639153177
column= defcontainer cutoff= 38.24107091745953 lower= 12.394642305841444 upper=
88.8767841407605
Identified outliers: 38
column= vending_y len= 15450 mean= 0.12987212038834953 std= 1.3552755036182966
column= vending_y cutoff= 4.06582651085489 lower= -3.9359543904665406 upper=
4.19569863124324
Identified outliers: 61
column= inputmethod.latin len= 15450 mean= 50.83265873851133 std=
13.056312844657608
column= inputmethod.latin cutoff= 39.16893853397282 lower= 11.663720204538507
upper= 90.00159727248415
Identified outliers: 34
column= dogalize_y len= 15450 mean= 49.48587180970874 std= 28.911527705274644
column= dogalize_y cutoff= 86.73458311582394 lower= -37.2487113061152 upper=
136.2204549255327
Identified outliers: 0
column= gapps len= 15450 mean= 49.9632538355987 std= 28.846959030294013
column= gapps cutoff= 86.54087709088203 lower= -36.577623255283335 upper=
136.50413092648074
Identified outliers: 0
column= simulator len= 15450 mean= 49.75620074951457 std= 28.925657829696828
column= simulator cutoff= 86.77697348909048 lower= -37.02077273957591 upper=
136.53317423860506
Identified outliers: 0
column= facebook_y len= 15450 mean= 49.80526438187702 std= 28.98326890126
column= facebook_y cutoff= 86.94980670378 lower= -37.144542321902975 upper=
136.75507108565702
Identified outliers: 0
column= google mobile services (gms) len= 15450 mean= 50.24647446407767 std=
28.84846131763429
column= google mobile services (gms) cutoff= 86.54538395290287 lower=
-36.298909488825196 upper= 136.79185841698055
Identified outliers: 0
column= google len= 15450 mean= 50.34342666990291 std= 28.79696748375251
column= google cutoff= 86.39090245125753 lower= -36.04747578135462 upper=
136.73432912116044
Identified outliers: 0
column= olauncher len= 15450 mean= 49.943178990291266 std= 29.069115788777484
column= olauncher cutoff= 87.20734736633246 lower= -37.26416837604119 upper=
137.15052635662371
Identified outliers: 0
column= browser provider len= 15450 mean= 49.762283370226534 std=
28.883676043571135
column= browser provider cutoff= 86.6510281307134 lower= -36.888744760486865
upper= 136.41331150093993
Identified outliers: 0
column= notifier len= 15450 mean= 49.63440627572815 std= 29.043983532101446
column= notifier cutoff= 87.13195059630434 lower= -37.49754432057618 upper=
136.76635687203247
Identified outliers: 0
column= gms.persistent len= 15450 mean= 49.77950399288026 std=
28.813966764780407
column= gms.persistent cutoff= 86.44190029434122 lower= -36.662396301460966
upper= 136.22140428722147
Identified outliers: 0
Total number of outliers is: 0
```

1.2 Data splitting

[135]:
```python
mwra = new_dataset[target_column]

data = new_dataset.drop(columns=[target_column], axis=1)

train_data, test_data, train_mwra, test_mwra = train_test_split(data, mwra,␣
↪test_size=test_size, random_state=random_state)
```

[136]:
```python
# features selected from our previous analysis
selected_features = ['gmail_x', 'gapps', 'facebook_x', 'chrome_x', 'vending_x',
                     'youtube', 'dogalize_x', 'updateassist', 'UCMobile.intl']

# this class was created with help from Claude.ai
# we were unsure of how to work with pipeline

class DataPreprocessor:
    def __init__(self):
        self.pipeline = None

    def create_pipeline(self):
        numeric_pipeline = Pipeline([
            ('standard_scaler', StandardScaler()),
            ('power_transform', PowerTransformer(method='yeo-johnson')),
            ('minmax_scaler', MinMaxScaler()),
            ('feature_select', SelectKBest(score_func=f_classif,␣
↪k=len(selected_features)))
        ])

        self.pipeline = ColumnTransformer(
            transformers=[
                ('numeric', numeric_pipeline, selected_features)
            ],
            remainder='drop'  # drop any columns not specified in the␣
↪transformers
        )

        return self

    def fit_transform(self, X, y=None):
        if self.pipeline is None:
            self.create_pipeline()
        return self.pipeline.fit_transform(X, y)

    def transform(self, X):
        if self.pipeline is None:
            raise ValueError("Pipeline has not been fitted yet. Call␣
↪fit_transform first.")
        return self.pipeline.transform(X)

def process_data(train_data, test_data, train_mwra=None):
    preprocessor = DataPreprocessor()

    X_train_processed = preprocessor.fit_transform(train_data, train_mwra)
    X_test_processed = preprocessor.transform(test_data)

    return X_train_processed, X_test_processed
```

[137]:
```
X_train_processed, X_test_processed = process_data(train_data, test_data,␣
↪train_mwra)

X_train_processed_df = pd.DataFrame(X_train_processed, ␣
↪columns=selected_features)
X_test_processed_df = pd.DataFrame(X_test_processed,  columns=selected_features)

X_train_processed_df.to_csv('dataset_df_train.csv', index=False)
X_test_processed_df.to_csv('dataset_df_test.csv', index=False)

print("Processed training data shape:", X_train_processed_df.shape)
print("Processed test data shape:", X_test_processed_df.shape)
X_train_processed_df
```

```
Processed training data shape: (10777, 9)
Processed test data shape: (4620, 9)
```

[137]:

```
        gmail_x     gapps  facebook_x  chrome_x  vending_x   youtube  \
0      0.427075  0.444816    0.341966  0.369723   0.510712  0.713017
1      0.781099  0.141603    0.592307  0.357581   0.626303  0.529742
2      0.482209  0.519524    0.410873  0.534186   0.501690  0.476516
3      0.547636  0.015084    0.500967  0.251712   0.541367  0.497678
4      0.319781  0.142447    0.574914  0.393006   0.267689  0.286099
…           …         …           …         …          …         …
10772  0.555393  0.700282    0.636286  0.557965   0.745899  0.342406
10773  0.483965  0.486749    0.683975  0.600363   0.779375  0.473029
10774  0.404798  0.737832    0.420983  0.499295   0.858299  0.407645
10775  0.623811  0.926622    0.708997  0.666297   0.713649  0.478073
10776  0.258322  0.887967    0.614210  0.506339   0.229566  0.337356

       dogalize_x  updateassist  UCMobile.intl
0        0.462090      0.421345       0.266698
1        0.213026      0.617507       0.531483
2        0.460207      0.457168       0.498022
3        0.632250      0.246029       0.593191
4        0.442534      0.695363       0.430522
…             …             …              …
10772    0.672724      0.155965       0.680182
10773    0.567626      0.391143       0.318646
10774    0.340618      0.520895       0.523320
10775    0.510748      0.420184       0.280551
10776    0.380861      0.324609       0.396991

[10777 rows x 9 columns]
```