Introduction to Isolation Forest

Salvatore Di Carlo

16/12/2019
Outline

• What is the Isolation forest technique:
  • Decision trees
  • Random forest
  • Isolation forest

• Why it is useful:
  • Example: wire scanner calibrations
  • Outliers detection

• How to use it:
  • Code snippet

• Conclusion
Decision trees

- Used for classification/regression.
- Learns a hierarchy of if/else tests.
- The N samples will be split until each leaf contains different values of the feature X.
- The argument generalizes to dataset containing several features $X_1, X_2, X_3, \ldots$
- The algorithms chooses the split that minimizes the variance within the leaves after the split.
- Random forests are ensembles of several (different) decision trees and the prediction is the average results.
The Isolation Forest technique

• The Isolation Forest is a technique for the detection of outlier samples.

• Since outliers have features X that differ significantly from most of the samples, they are isolated earlier in the hierarchy of a decision tree.

• Outliers are detected by setting a threshold on the mean length (number of splits) from the top of the tree downwards.

• The Scikit-learn implementation provides a score for each sample that increases from -1 to +1 with the number of splits.
Wire scanners are calibrated by using a laser in order to transform the angular coordinate $\alpha$ into the cartesian coordinate $y$.

**Polynomial fitting**

$$y' = x_0 + x_1 \alpha + \ldots + x_5 \alpha^5$$

**Sinusoidal fitting**

$$y' = a + b \sin(\beta)$$

$\beta = c\alpha + d$

$b$ represents the fork length
Outliers detection

• The algorithm is applied to our dataset of calibrations and returns a score for each sample.

• The sample with lower score are likely to be outliers.

• The outlier threshold on the score must be set by the user.

• In our case, we can set the threshold in a way that the parameter b from the fitting for the inliers (total dataset without the outliers) is reasonably close to the nominal fork length.
The initial dataset contained 242 samples;

Choosing a threshold score of 0.04 provides 18 candidate outliers;

The mean of each feature for the inliers is significantly different from the corresponding value for the outliers;

b (fork length) can should be either about 150 mm or about 180 mm.

### Inliers/Outliers comparison

<table>
<thead>
<tr>
<th>Av.Speed [rads-1]</th>
<th>Scan</th>
<th>Serial</th>
<th>OUTLIERS</th>
<th>INLIERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>126</td>
<td>IN PSB_PXBSRA005-CR000002</td>
<td>-6.940890</td>
<td>45.156000</td>
</tr>
<tr>
<td></td>
<td>120</td>
<td>OUT PSB_PXBSRA005-CR000007</td>
<td>-7.276314</td>
<td>44.814874</td>
</tr>
<tr>
<td>66</td>
<td>123</td>
<td>IN PSB_PXBSRA005-CR000006</td>
<td>56.274344</td>
<td>117.166833</td>
</tr>
<tr>
<td></td>
<td>53</td>
<td>IN PSB_PXBSRA005-CR0000007</td>
<td>12.296224</td>
<td>57.898795</td>
</tr>
<tr>
<td></td>
<td>53</td>
<td>IN PSB_PXBSRA005-CR0000044</td>
<td>-16.377713</td>
<td>36.281998</td>
</tr>
<tr>
<td></td>
<td>102</td>
<td>IN PSB_PXBSRA005-CR0000002</td>
<td>-21.590523</td>
<td>37.521772</td>
</tr>
<tr>
<td></td>
<td>112</td>
<td>IN PSB_PXBSRA005-CR0000007</td>
<td>-21.127997</td>
<td>37.152064</td>
</tr>
<tr>
<td>102</td>
<td>59</td>
<td>OUT PSB_PXBSRA005-CR0000007</td>
<td>-16.379092</td>
<td>36.229681</td>
</tr>
<tr>
<td></td>
<td>106</td>
<td>OUT PSB_PXBSRA005-CR0000002</td>
<td>-21.563820</td>
<td>37.374712</td>
</tr>
<tr>
<td></td>
<td>122</td>
<td>OUT PSB_PXBSRA005-CR0000007</td>
<td>-21.165705</td>
<td>37.527720</td>
</tr>
<tr>
<td>122</td>
<td>110</td>
<td>IN PSB_PXBSRA005-CR0000001</td>
<td>100.594327</td>
<td>179.214612</td>
</tr>
<tr>
<td></td>
<td>57</td>
<td>OUT PSB_PXBSRA005-CR0000000</td>
<td>97.175192</td>
<td>175.513530</td>
</tr>
<tr>
<td></td>
<td>107</td>
<td>OUT PSB_PXBSRA005-CR0000001</td>
<td>112.164005</td>
<td>191.693267</td>
</tr>
<tr>
<td></td>
<td>115</td>
<td>OUT PSB_PXBSRA005-CR0000001</td>
<td>102.030585</td>
<td>180.732285</td>
</tr>
<tr>
<td></td>
<td>115</td>
<td>IN PSB_PXBSRA005-CR0000003</td>
<td>95.099272</td>
<td>173.410579</td>
</tr>
<tr>
<td></td>
<td>127</td>
<td>OUT PSB_PXBSRA005-CR0000000</td>
<td>96.537273</td>
<td>173.922995</td>
</tr>
<tr>
<td></td>
<td>111</td>
<td>OUT PSB_PXBSRA005-CR0000007</td>
<td>126.051426</td>
<td>207.098798</td>
</tr>
<tr>
<td></td>
<td>112</td>
<td>OUT PSB_PXBSRA005-CR0000004</td>
<td>97.445557</td>
<td>175.353764</td>
</tr>
</tbody>
</table>
Main model parameters:

- behaviour: this parameter has no effect, is deprecated, and will be removed;
- bootstrap: randomly re-sample initial sample with replacement;
- contamination: % of outliers (if known), no effect on score;
- max_features: # of randomly selected features at each node;
- max_samples: # of samples to draw from the initial samples for each tree;
- n_estimators: # of trees in the forest;
- n_jobs: # of cores used for parallelization, -1 means max available;
- random_state: sets the seed for reproducibility;
- verbose: controls the verbosity of the output;
- warm_start: reuse the solution from the previous call and add more trees to the old ensemble.

Code snippet

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as tck

from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import IsolationForest

df = pd.read_csv('calibration_table_cos.csv')
df = df.drop('Unnamed: 0', 1)
X = df.copy()

X['Serial_Scan'] = X['Serial'] + '_' + X['Scan']
cat_cols = ['Scan', 'Serial', 'Serial_Scan']
label_encoder = LabelEncoder()

for col in cat_cols:
    X[col] = label_encoder.fit_transform(X[col])

if_model = IsolationForest(behaviour='new',
                          bootstrap=True,
                          contamination=0.01,
                          max_features=1.0,
                          max_samples=len(X),
                          n_estimators=100,
                          n_jobs=-1,
                          random_state=1,
                          verbose=0,
                          warm_start=False)

if_model.fit(X)

y_pred = if_model.predict(X)
y_scores = if_model.decision_function(X)

outliers_index = np.where(y_scores < 0.04)[0]
df_inliers = df[~df.index.isin(outliers_index)]
df_outliers = df[df.index.isin(outliers_index)]
```
Decision trees/forests provide an intuitive and useful tool in machine learning that can be used both for data classification and regression.

The isolation forest technique is useful to isolate outliers, i.e., samples that significantly differ from the rest of the data.

As an example we isolated some outlier using the wire scanner calibrations dataset, and the corresponding code snippet was provided.

A certain degree of arbitrariness remains in setting the threshold score for outliers. Plotting a histogram of the score can help in deciding the optimal value.