In our experiments, we determined that the value of is quite crucial in mounting a successful frog boiling attack. The idea behind the attack is to pass off forged banknotes as authentic, so, we should try to get the classifier to determine as many forged notes in the test data set as possible, as authentic. For this reason, for the remainder of this paper, when we refer to the authentic data set (Adata), it refers to the records in the training set that were classified as authentic, and when we refer to the forged data set (Fdata), it refers to the records in the test set that were classified as forged.

In each iteration, we start off by calculating the mean (centroid) of all values in Fdata, and find the point in Adata that’s closest to the centroid, let’s call it CAD (closest authentic data point). We then generate a new fake data point (FDP) by adding a tiny incremental value to the CAD. This incremental value is given by the formula

distance between CAD and Centroid of Fdata

number of iterations

We then add this new FDP back into Adata, manually assign it a classification of authentic, refit the model, and predict the classification of the FDP. If the classification is forged, we stop right there (as our attack is compromised). If the classification is authentic, we proceed to the next iteration.

With the above-mentioned approach, we were able to successfully ‘attack’ the system into incorrectly predicting a forged banknote as authentic. We ran the algorithm against five different classifiers, and some of them displayed a stronger aversion to false detection than the others, but with enough iterations, we were able to crack them all eventually (with the exception of the Random Forest classifier). The tables below detail the results, with the highlighted row indicating where we saw a drop in the accuracy.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Random Forest | | | | | | | | | |
| Training Set | | | | | Test Set | | | | |
| Count | | Correctly Predicted | | Accuracy | Count | | Correctly Predicted | Accuracy | |
| Initial Run | 1097 | | 1097 | | 100.00% | 275 | | 274 | 99.64% | |
| After 50 iterations | 1106 | | 1105 | | 99.91% | 275 | | 274 | 99.64% | |
| After 100 iterations | 1144 | | 1144 | | 100.00% | 275 | | 274 | 99.64% | |
| After 200 iterations | 1135 | | 1135 | | 100.00% | 275 | | 275 | 100.00% | |
| After 500 iterations | 1219 | | 1219 | | 100.00% | 275 | | 275 | 100.00% | |
| After 1000 iterations | 1339 | | 1339 | | 100.00% | 275 | | 275 | 100.00% | |
| After 2000 iterations | 1458 | | 1458 | | 100.00% | 275 | | 275 | 100.00% | |
| After 5000 iterations | 2470 | | 2470 | | 100.00% | 275 | | 274 | 99.64% | |
| After 10000 iterations | 3859 | | 3858 | | 99.97% | 275 | | 275 | 100% | |
|  | | Linear SVC | | | | | | | |
| Training Set | | | | | Test Set | | |
| Count | | Correctly Predicted | Accuracy | | Count | Correctly Predicted | Accuracy |
| Initial Run | | 1097 | | 1088 | 99.18% | | 275 | 271 | 98.55% |
| After 50 FBA iterations | | 1108 | | 1096 | 98.92% | | 275 | 270 | 98.18% |
| After 100 FBA iterations | | 1123 | | 1106 | 98.49% | | 275 | 268 | 97.45% |
| After 200 FBA iterations | | 1167 | | 1150 | 98.54% | | 275 | 268 | 97.45% |
| After 500 FBA iterations | | 1363 | | 1333 | 97.80% | | 275 | 266 | 96.73% |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Neural Network | | | | | |
| Training Set | | | Test Set | | |
| Count | Correctly Predicted | Accuracy | Count | Correctly Predicted | Accuracy |
| Initial Run | 1097 | 1097 | 100.00% | 275 | 275 | 100.00% |
| After 50 FBA iterations | 1112 | 1112 | 100.00% | 275 | 275 | 100.00% |
| After 100 FBA iterations | 1197 | 1197 | 100.00% | 275 | 274 | 99.64% |
| After 200 FBA iterations | 1297 | 1297 | 100.00% | 275 | 274 | 99.64% |
| After 500 FBA iterations | 1597 | 1597 | 100.00% | 275 | 274 | 99.64% |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Gaussian | | | | | |
| Training Set | | | Test Set | | |
| Count | Correctly Predicted | Accuracy | Count | Correctly Predicted | Accuracy |
| Initial Run | 1097 | 1094 | 98.55% | 275 | 273 | 99.27% |
| After 50 FBA iterations | 1117 | 1111 | 99.46% | 275 | 273 | 99.27% |
| After 100 FBA iterations | 1154 | 1148 | 99.48% | 275 | 270 | 98.18% |
| After 200 FBA iterations | 1297 | 1286 | 99.15% | 275 | 269 | 97.82% |
| After 500 FBA iterations | 1597 | 1578 | 98.81% | 275 | 267 | 97.09% |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | KNN | | | | | |
| Training Set | | | Test Set | | |
| Count | Correctly Predicted | Accuracy | Count | Correctly Predicted | Accuracy |
| Initial Run | 1097 | 1096 | 99.91% | 275 | 275 | 100.00% |
| After 50 FBA iterations | 1147 | 1145 | 99.83% | 275 | 274 | 99.64% |
| After 100 FBA iterations | 1197 | 1193 | 99.67% | 275 | 274 | 99.64% |
| After 200 FBA iterations | 1297 | 1293 | 99.69% | 275 | 274 | 99.64% |
| After 500 FBA iterations | 1597 | 1593 | 99.75% | 275 | 274 | 99.64% |

There were several noteworthy observations that we could make.

1. With the Neural Network and the KNN classifiers, since we drifted Adata towards the mean of Fdata, post the attack, it was only the data point closest to the mean that was classified as authentic. For the attack to succeed on several data points, we need to run it multiple times, once each towards the forged datapoint we want the system to misclassify.
2. With the Gaussian and the LinearSVC classifiers, with increasing iterations, we were able to classify more of Fdata as authentic.
3. The LinearSVC model chosen by the original researchers performed the worst, with as many as 9 mispredictions post 500 iterations of the attack.
4. The Random Forest classifier really surprised us by almost always thwarting the attack at around the 30% mark of the number of iterations. The initial count for the number of data points in the Adata is 1097. For classifiers like KNN (that never detected the “corrupt” data points we were introducing, as forged) you can see that the end of n iterations, the number of data points in the Adata set had increased to 1097 + n. Whereas in the case of Random Forest, even setting the iteration count to as high as 10000 only resulted in the classifier detected the corrupt data point after ~2700 iterations. Not only that, but its test and training accuracy seemed to get better the more iterations we ran.
5. We were able to reproduce the attack by starting with the mean value of the Adata, rather than choosing the point closest to the Fdata, but it required significantly higher number of iterations (20x) to achieve the same result.

Abstract changes:

We conclude by investigating potential vulnerabilities an online machine learning system can encounter, specifically the Frog Boiling Attack. We outline a potential implementation of this type of penetration attack. In our experiments, depending on the classifier that was chosen, we were able to get the system to successfully mis-predict a forged data point as authentic in as few as 50 iterations. The complexity of the classifier didn’t offer better protection against the attack. We saw that the simple k-Nearest-Neighbor performed better than the Support Vector Machine & Gaussian models.

We also highlight various potential countermeasures to mitigate its negative impact on a machine learning system in production.