Poisoning_Attack_Spam_Filter

April 18, 2022

1 Poisoning Attack Spam Filter

Code based on [Build a machine learning email spam detector with Python](https://blog.logrocket.com/email-spam-detector-python-machine-learning/

1.1 Import Libraries

[102]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn import svm
```

1.2 Download Data and Process CSV

[103]:
```
!wget https://raw.githubusercontent.com/SmallLion/Python-Projects/main/
↪Spam-detection/spam.csv
```

```
--2022-04-18 20:17:07--  https://raw.githubusercontent.com/SmallLion/Python-
Projects/main/Spam-detection/spam.csv
Resolving raw.githubusercontent.com (raw.githubusercontent.com)…
185.199.108.133, 185.199.109.133, 185.199.110.133, …
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|185.199.108.133|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 499362 (488K) [text/plain]
Saving to: ‘spam.csv.1’

spam.csv.1          100%[===================>] 487.66K  --.-KB/s    in 0.04s

2022-04-18 20:17:07 (13.1 MB/s) - ‘spam.csv.1’ saved [499362/499362]
```

[104]:
```
spam = pd.read_csv('spam.csv')
```

1.3 Split Dataset into Train and Test

[136]:
```
x = spam['v2']
y = spam["v1"]
x_train, x_test, y_train, y_test = train_test_split(z,y,test_size = 0.2)
```

1.4 Obtain Features and Train Model

[137]:
```
cv = CountVectorizer()
features = cv.fit_transform(x_train)
```

[138]:
```
model = svm.SVC()
model.fit(features,y_train)
```

[138]:
```
SVC()
```

1.5 Evaluate Model Accuracy

[139]:
```
features_test = cv.transform(x_test)
print("Accuracy: {}".format(model.score(features_test,y_test)))
```

```
Accuracy: 0.9847533632286996
```

1.6 Poison 50% of Training Data and Re-train model

[140]:
```
import random

for i, y in enumerate(y_train):
  if random.random() > 0.5:
    y_train.iloc[i] = 'spam'
```

[141]:
```
model = svm.SVC()
model.fit(features,y_train)
```

[141]:
```
SVC()
```

1.7 Evaluate Poisoned Model Accuracy

[142]:
```
features_test = cv.transform(x_test)
print("Accuracy: {}".format(model.score(features_test,y_test)))
```

```
Accuracy: 0.42152466367713004
```