Enhancing Mobile Security through Machine Learning: A Study on Android Malware Detection

Authors:
Moti Dahari308212570
amit koobani 204804488

Machine Learning-based Approach for Android Malware Detection: An Evaluation of Performance and Limitations

Introduction

המצגת שלנו מכסה גישה מבוססת למידת מכונה לאיתור תוכנות זדוניות במכשירי אנדרואיד. השתמשנו במערך נתונים של 4591 אפליקציות והכשרנו עליו מודלים שונים. נדון במתודולוגיה, בתוצאות, במדדי הערכה ובתובנות, כמו גם במגבלות וכיוונים עתידיים למחקר זה.

Description of the dataset

בפרויקט זה השתמש במערך נתונים של 4591 אפליקציות, במהלך חילוץ הנתונים נבחרו 1817 אפליקציות, 165 תויגו כזדוניות ו- 1652 כשפירות.

בנינו מודלים ,jsonלפני העברת המידע למודל חילצנו את הנתונים לתוך קובץ

שוני של למידת מכונה כמו:

עם ליבה לינארית SVM

Logistic Regression

GradientBoostingClassifier

DecisionTreeClassifier

KNeighborsClassifier

מערך הנתונים כלל קטגוריות שונות של פיצרים מסוגים שונים.

JSONהמידע שחולץ מהאפליקציות אוחסן בקובץ

כל הערכים החסרים מולאו ב-0(הפיצרים), בנוסף נכתבו פונקציות עזר כדי לחלץ ולארגן את המידע.

Pre-processing

עיבדנו מראש את הנתונים כדי להכין אותם לאימון המודל. השלבים כללו ניקוי נתונים, טרנספורמציה, הפחתת מימד ואיזון מערך הנתונים. זה היה הכרחי כדי להבטיח שהנתונים יהיו באיכות גבוהה וכדי להתגבר על האתגרים הניצבים בפני הפרויקט.

Details of the features

פרויקט זה חילץ תכונות שונות ממערך הנתונים של אפליקציות אנדרואיד:

```
{
    sha256,
    label,
    app_permissions,
    api_permissions,
    api_calls,
    activities,
    s_and_r,
    interesting_calls,
    urls,
    providers
}
```


```
"sha256": {"malicious": 2755,"benign": 1836},
"interesting_calls::getSystemService": {"malicious": 2563,"benign": 1334},
"app_permissions::android_permission_INTERNET": {"malicious": 2510,"benign": 1423},
"api_permissions::android_permission_INTERNET": {"malicious": 2460,"benign": 1306},
"app_permissions::android_permission_READ_PHONE_STATE": {"malicious": 2353,"benign": 286},
"api_permissions::android_permission_READ_PHONE_STATE": {"malicious": 2079,"benign": 330},
"api_calls::java/net/URL;->openConnection": {"malicious": 1864,"benign": 951},
"interesting_calls::getDeviceId": {"malicious": 1857,"benign": 446},
"api_calls::android/telephony/TelephonyManager;->getDeviceId": {"malicious": 1856,"benign": 238},
"interesting_calls::printStackTrace": {"malicious": 1840,"benign": 654},
"api_calls::java/net/HttpURLConnection": {"malicious": 1804,"benign": 932},
"app_permissions::android_permission_ACCESS_NETWORK_STATE": {"malicious": 1778,"benign": 9
"app_permissions::android_permission_WRITE_EXTERNAL_STORAGE": {"malicious": 1776,"benign": 509},
"api_calls::android/webkit/WebView": {"malicious": 1720,"benign": 1049},
"api_calls::android/content/Context;->startService": {"malicious": 1710,"benign": 682},
"api_calls::org/apache/http/impl/client/DefaultHttpClient": {"malicious": 1668,"benign": 623},
"api_permissions::android_permission_VIBRATE": {"malicious": 1612,"benign": 722},
"api_calls::android/content/Context;->startActivity": {"malicious": 1576,"benign": 1001},
"api_calls::android/app/NotificationManager;->notify": {"malicious": 1535,"benign": 637},
"api_permissions::android_permission_ACCESS_NETWORK_STATE": {"malicious": 1508,"benign": 1086},
"api_calls::java/net/HttpURLConnection;->connect": {"malicious": 1474,"benign": 679},
"interesting_calls::HttpPost": {"malicious": 1437,"benign": 492},
"app_permissions::android_permission_SEND_SMS": {"malicious": 1417,"benign": 35},
"api_calls::android/net/ConnectivityManager;->getActiveNetworkInfo": {"malicious": 1411,"benign": 1008},
"interesting_calls::Read/Write External Storage": {"malicious": 1376,"benign": 782},
"interesting_calls::getPackageInfo": {"malicious": 1333,"benign": 1317},
```

```
[
  {
    "sha256": "B8CC23EC7D68F320558A9572F1C14935AFC820A0A1C053ECA94888F341D208DF",
    "label": 1,
    "app_permissions::android_permission_READ_PHONE_STATE": 1,
    "app_permissions::android_permission_SEND_SMS": 1,
    "app_permissions::android_permission_RECEIVE_SMS": 1,
    "app_permissions::android_permission_INTERNET": 1,
    "api_permissions::android_permission_INTERNET": 1,
    "api_permissions::android_permission_READ_PHONE_STATE": 1,
    "api_permissions::android_permission_SEND_SMS": 1,
    "api_calls::android/app/Activity;->startActivity": 1,
    "api_calls::android/app/Activity;->startActivityForResult": 1,
    "api_calls::org/apache/http/impl/client/DefaultHttpClient": 1,
    "api_calls::org/apache/http/impl/client/DefaultHttpClient;->execute": 1,
    "api_calls::android/telephony/TelephonyManager;->getLine1Number": 1,
    "api_calls::android/telephony/SmsManager;->sendTextMessage": 1,
    "activities::b'_FirstActivity'": 1,
    "activities::b'_RulesActivity'": 1,
    "activities::b'_FinishActivity'": 1,
    "activities::b'QuestionActivity'": 1,
    "activities::b'MemberActivity'": 1,
    "s_and_r::b'_services_SMSSenderService'": 1,
    "s_and_r::b'_sms_BinarySMSReceiver'": 1,
    "interesting_calls::printStackTrace": 1,
    "interesting_calls::getSystemService": 1,
    "interesting_calls::getSimCountryIso": 1,
    "interesting_calls::sendSMS": 1
  },
```

Stages to work on this project:

לפרויקט היו מספר שלבים:

1. איסוף נתונים
2. מיצוי תכונות
3. עיבוד מקדים של נתונים הדרכת מודלים
4. הערכה.

התכונות שימשו לאימון מודלים של למידת מכונה כגון :

SVM with a linear kernel
Logistic Regression
KNeighborsClassifier
DecisionTreeClassifier
GradientBoostingClassifier

ביצועי המודלים הוערכו באמצעות מדדים כגון דיוק, דיוק וזכירה.
נמצא כי הגישה המוצעת עולה על השיטות המסורתיות במונחים של דיוק, דיוק וזכירה, אך עדיין יש מקום לשיפור. כמו כן, כתבנו סקריפטים שונים ופונקציות עוזר לאורך כל הפרויקט כדי להבטיח ביצוע חלק.

Information fragmentation

לסיכום, פיצול מידע שימש בפרויקט זה כדי לפרק מערך נתונים גדול של יישומי אנדרואיד לנתחים קטנים יותר וניתנים לניהול. זה עזר לצוות לחלץ בקלות תכונות רלוונטיות, לבצע ניתוח מעמיק ולאחסן ולאחזר נתונים ביעילות.
זה מילא תפקיד מכריע בהפיכת הנתונים לשימושים לאימון מודל למידת המכונה.

פיצול מידע היה מכריע בפרויקט זה מכיוון שהוא עזר לצוות לפרק את מערך הנתונים הגדול של 1817 יישומי אנדרואיד לנתחים קטנים יותר לניהול, ניתוח ואחסון טובים יותר. סקריפטים ופונקציות עוזר נכתבו כדי לסייע בחילוץ נתונים ועיבוד מקדים, המאפשרים ניתוח ממוקד של היבטים ספציפיים כגון קטגוריות וקריאות API. זה עזר לזהות דפוסים ומגמות רלוונטיות לזיהוי תוכנות זדוניות, ושיפר את הביצועים של מודלים של למידה חישובית. באופן כללי, לפרגמנטציה היה תפקיד מפתח בעבודה יעילה עם הנתונים ובהשגת תוצאות מדויקות.

Algorithms

בפרויקט זה, נעשה שימוש באלגוריתמים שונים של למידת מכונה כדי להכשיר מודלים לזיהוי תוכנות זדוניות באנדרואיד, כולל:
SVM, Logistic Regression, KNeighborsClassifier, DecisionTreeClassifier ,-ו
GradientBoostingClassifier.
לכל אלגוריתם היו נקודות החוזק והחולשה שלו, יש צורך במחקר נוסף כדי לשפר את הדיוק והביצועים של השיטה המוצעת.

Algorithms

GradientBoostingClassifier

להוסיף הסבר

KNeighborsClassifier

להוסיף הסבר

LinearSVC

להוסיף הסבר

DecisionTreeClassifier

להוסיף הסבר

LogisticRegression

Challenges

התמודדנו עם מספר אתגרים בפרויקט, כולל התמודדות עם מערך נתונים גדול, נתונים מורכבים ואיכות נתונים. הם גם נאלצו להשתמש בטכניקות הפחתת מימד כדי לטפל בממדיות גבוהה, ולהעריך בקפידה אלגוריתמים שונים כדי לקבוע איזה מהם מתאים ביותר למשימה.

Assumptions

עשינו מספר הנחות כדי להשלים את המשימה של זיהוי תוכנות זדוניות באנדרואיד, כמו הנחה שמערך הנתונים היה מייצג, מסומן שבהם נעשה שימוש מתאימים. עם זאת, ייתכן שהנחות אלו MLכהלכה, התכונות היו רלוונטיות ואינפורמטיביות, ואלגוריתמי ה-אינן נכונות ועלולות להוביל למודלים לא מדויקים.

links

github :https://github.com/motidahari/Mobile-Security-ML-Android-Malware-Detection

dataset: https://github.com/motidahari/Mobile-Security-ML-Android-Malware-Detection/tree/main/data/apks/result

result algorithms: https://github.com/motidahari/Mobile-Security-ML-Android-Malware-Detection-Machine-learning-course/blob/main/data/apks/result/bestValuesForAlgoritems.json