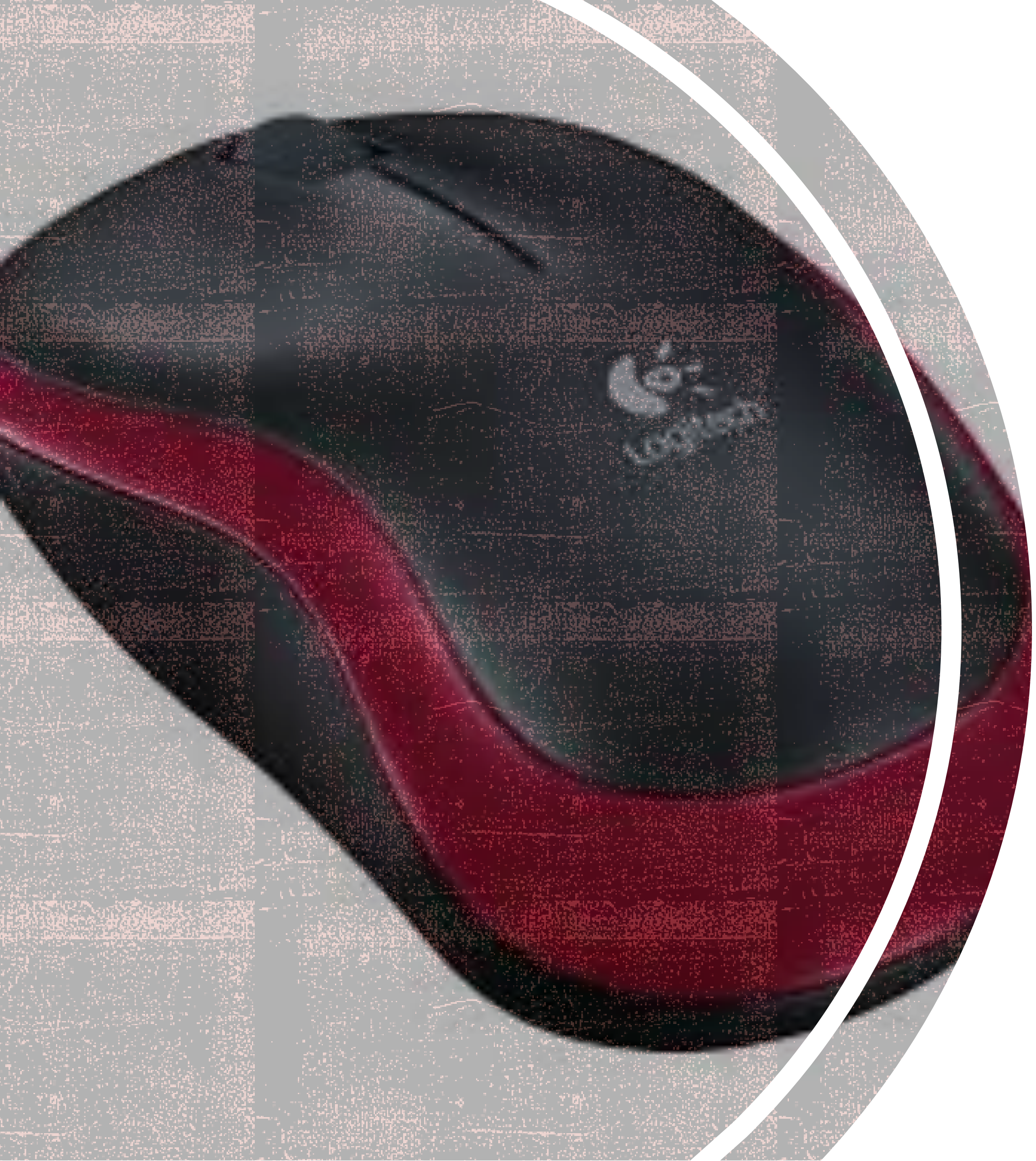


MOUSE MOVEMENT ANALTICS ON SCALA + SPARK

Eszter Windhager-Pokol

AGENDA

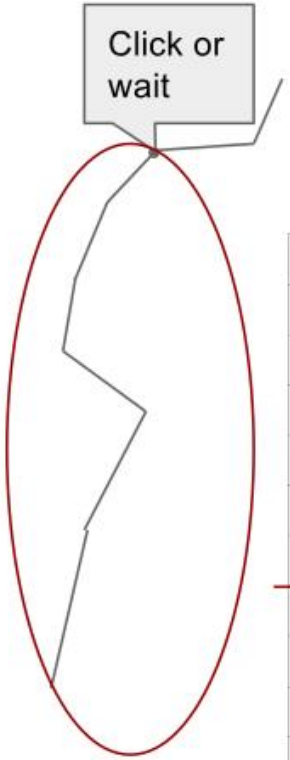
- **Topic**: mouse movement analytics in IT security
- **Issue**: production environment (scala + Spark)
- **Solution**: H2O

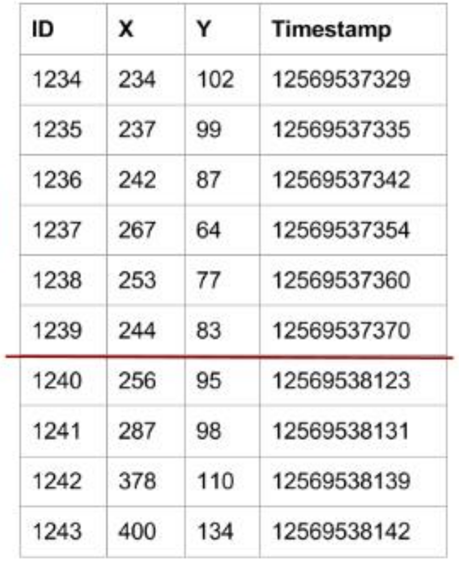
MOUSE MOVEMENT ANALYTICS IN IT SECURITY

- Machine learning driven IT security product
- Create profiles for users based on logs and audit trails
- Examine all aspects of behavior

MOUSE MOVEMENT ANALYTICS

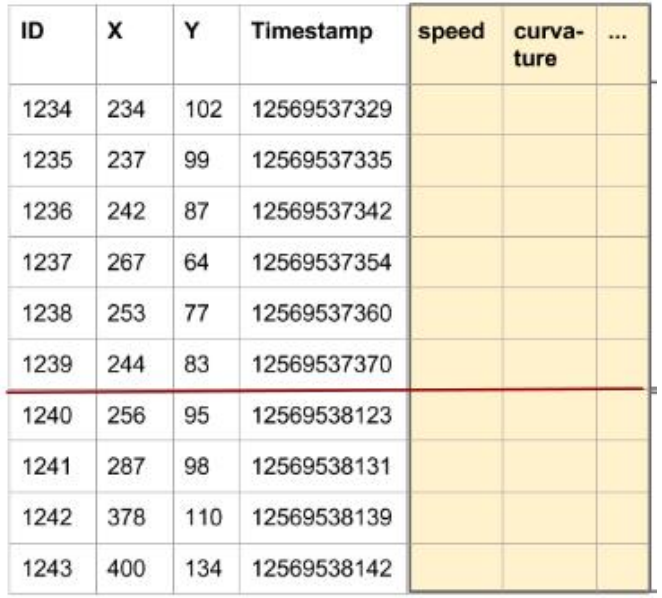
Separate gestures


Click or wait



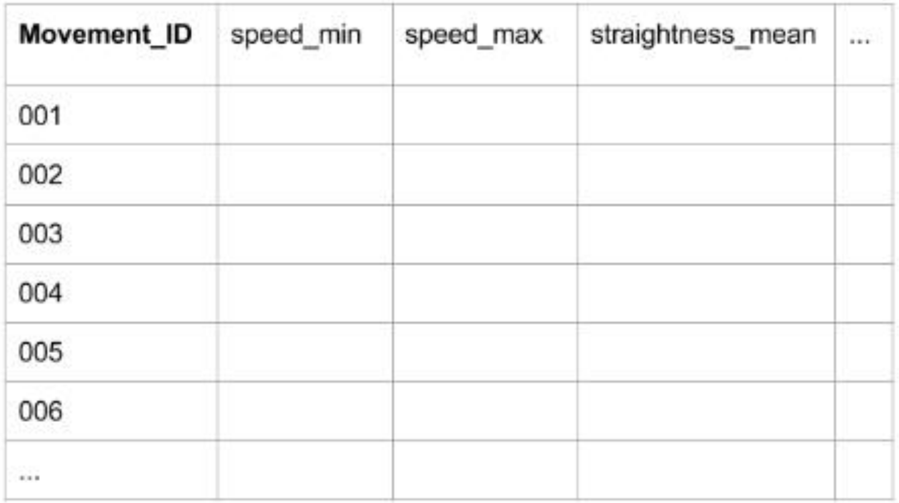
ID	X	Y	Timestamp
1234	234	102	12569537329
1235	237	99	12569537335
1236	242	87	12569537342
1237	267	64	12569537354
1238	253	77	12569537360
1239	244	83	12569537370
1240	256	95	12569538123
1241	287	98	12569538131
1242	378	110	12569538139
1243	400	134	12569538142

Descriptive statistics



ID	X	Y	Timestamp	speed	curva-ture	...
1234	234	102	12569537329			
1235	237	99	12569537335			
1236	242	87	12569537342			
1237	267	64	12569537354			
1238	253	77	12569537360			
1239	244	83	12569537370			
1240	256	95	12569538123			
1241	287	98	12569538131			
1242	378	110	12569538139			
1243	400	134	12569538142			

Aggregate to gestures



Movement_ID	speed_min	speed_max	straightness_mean	...
001				
002				
003				
004				
005				
006				
...				

MOUSE MOVEMENT ANALYTICS

- Divided into gestures
- Features: speed, acceleration, curvature, straightness, angle speed …
- Labels: mouse/touchpad, users
- Differentiate between user A and everyone else
- GBM was the best

PRODUCTION ENVIRONMENT

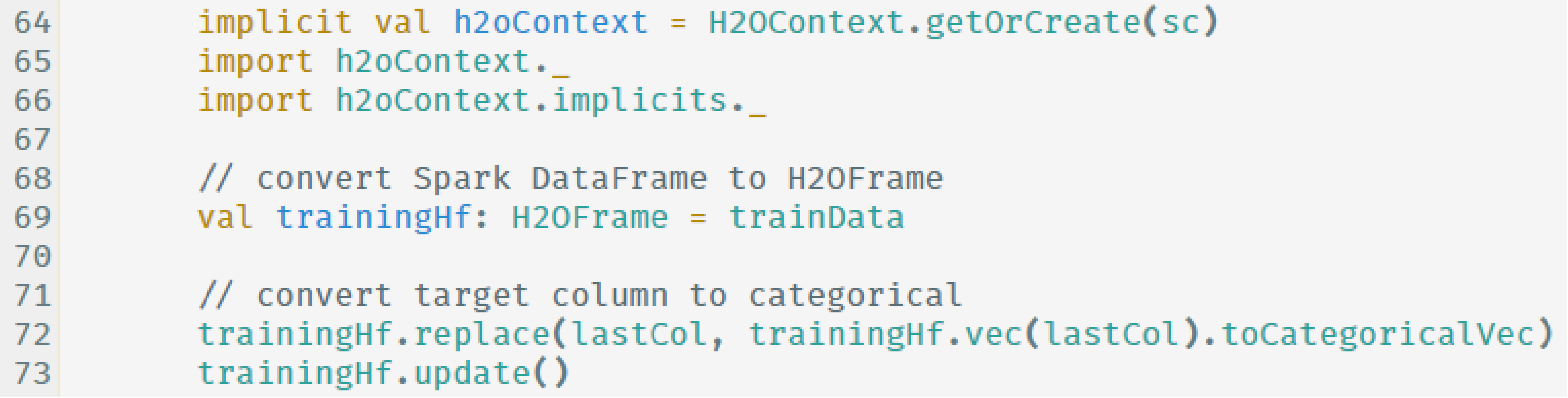
- Scala + Spark

- Spark ML – requires special format DataFrame

Target	Input vector
1	[3.54, -2.3, 0.018, 45.42, 354.5, 23.1, 232, 2, 34.1, -11.01, 78.02, …]
0	[8.11, 1.5, 0.045, 42.45, 597.4, 18.1, 321, 5, 37.1, -27.34, 87.21, …]
…	…

PRODUCTION ENVIRONMENT

H2O



```
        implicit val h2oContext = H2OContext.getOrCreate(sc)
        import h2oContext._
        import h2oContext.implicits._

        // convert Spark DataFrame to H2OFrame
        val trainingHf: H2OFrame = trainData

        // convert target column to categorical
        trainingHf.replace(lastCol, trainingHf.vec(lastCol).toCategoricalVec)
        trainingHf.update()
```

SOLUTION: H2O

- Easy to use
- Few lines of code
- Additional benefit: pojo/mojo extract
- Spark not needed for scoring