Model Extraction Attacks

Presentation By Abbigail Waddell, *North Carolina A&T State University*

Based on the Paper

Stealing Machine Learning Models via Prediction APIs

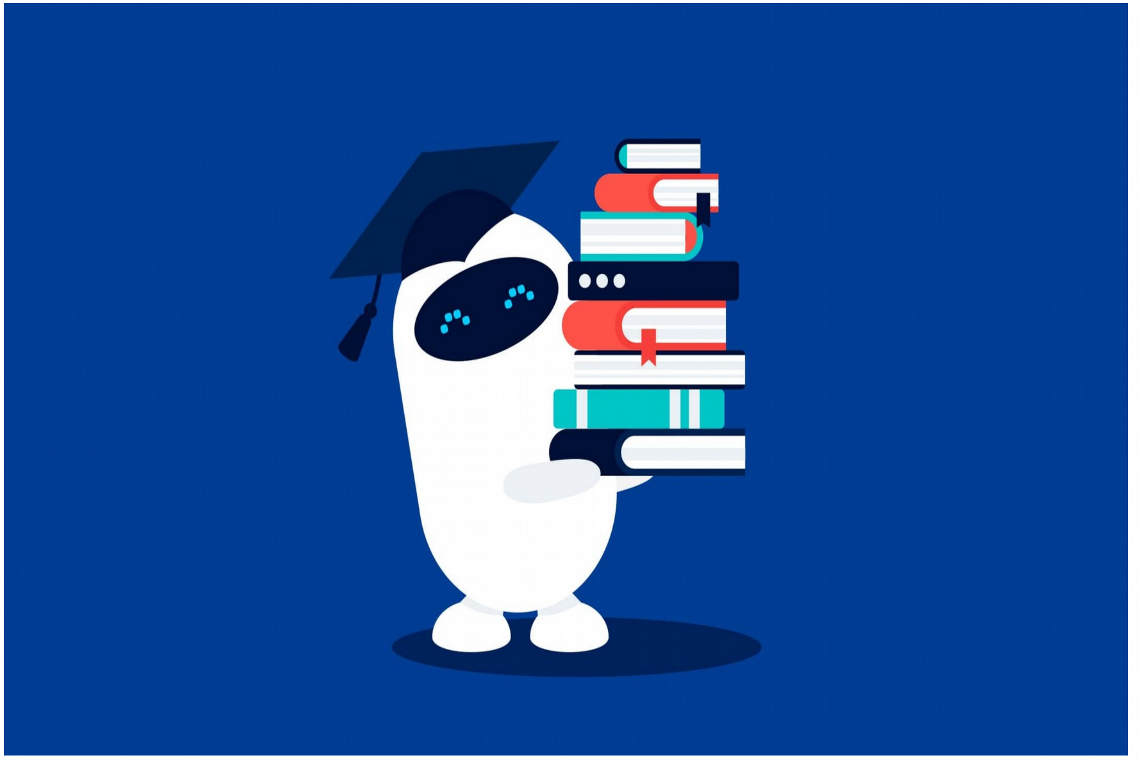
Authors: Florian Tramèr, *École Polytechnique Fédérale de Lausanne*; **Fan Zhang,** *Cornell University*; **Ari Juels,** *Cornell Tech*;

Michael K. Reiter, *The University of North Carolina at Chapel Hill*; **Thomas Ristenpart,** *Cornell Tech*

Presented at the 25th USENIX Security Symposium

https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer

Overview



Machine Learning Systems

1. Gather labeled data

$x^{(n)}$ n-dimensional feature vector x (data) 

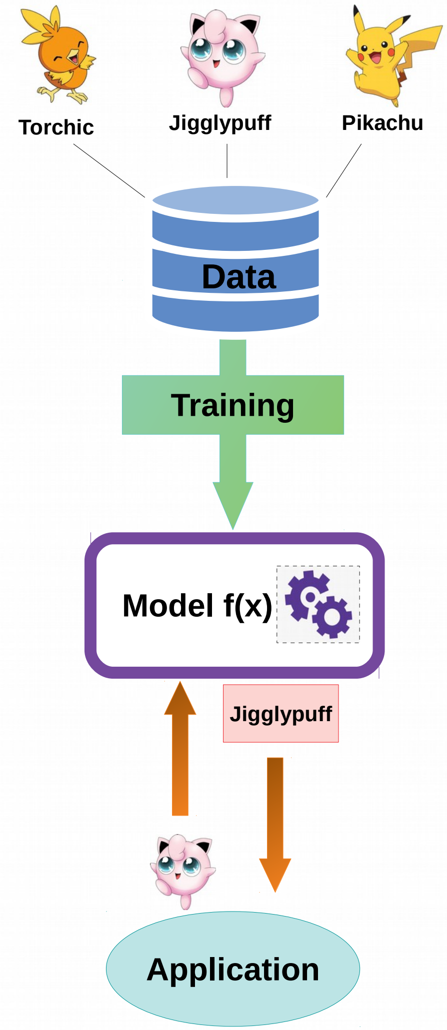
$y^{(n)}$ dependent variable y (labels) Torchic

2. Train model f(x) using the labeled data

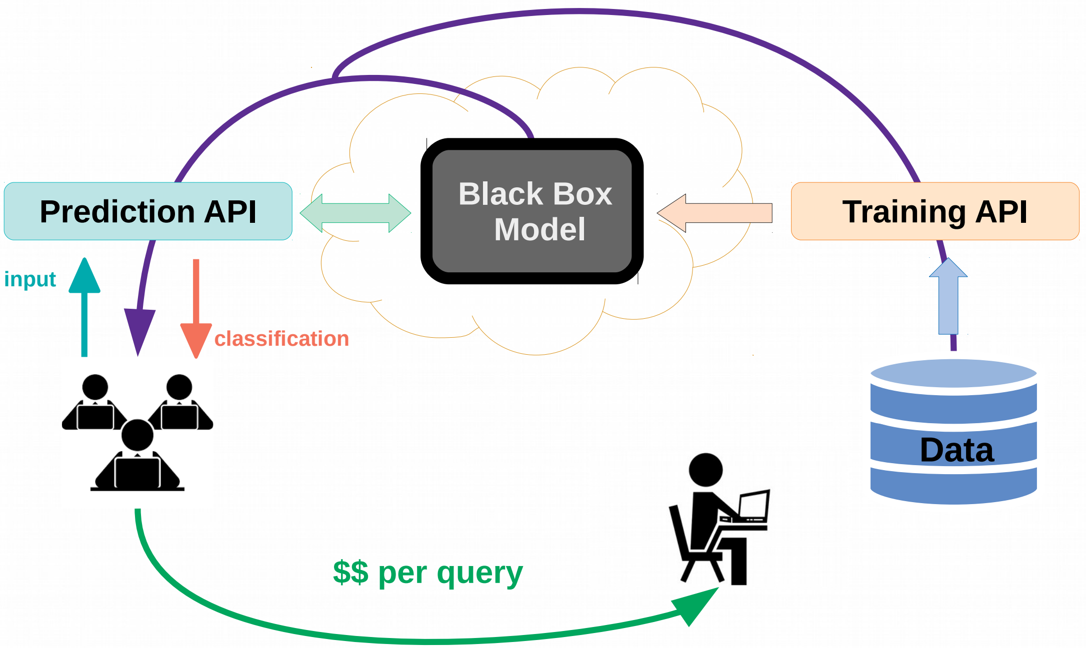
f(x) = y — Prediction

Confidence

3. Use model f(x) in application or publish for others to use



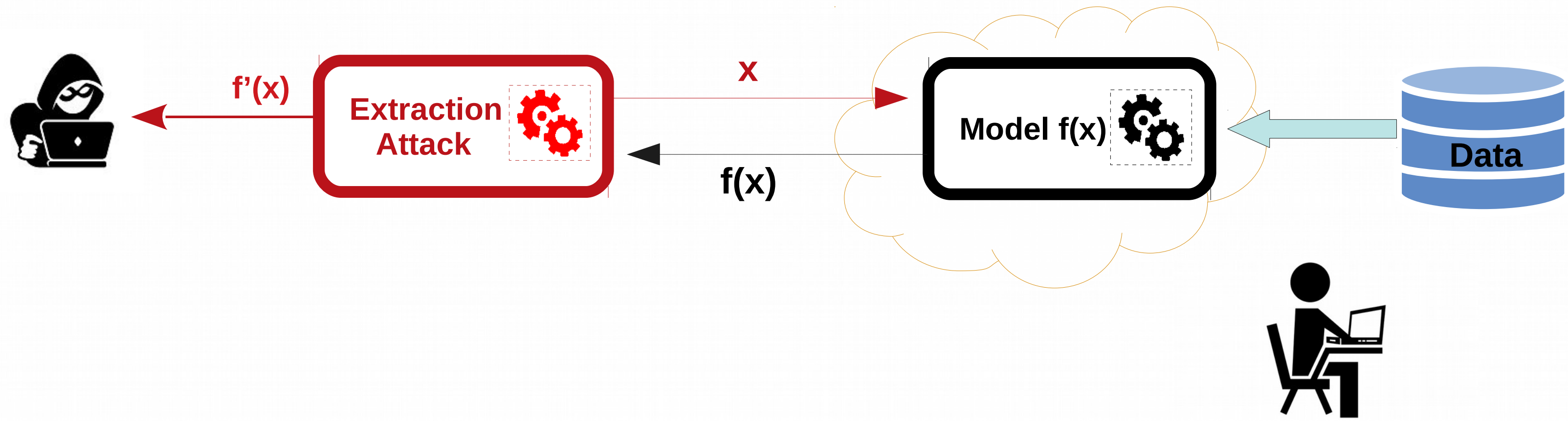
Machine Learning As a Service



Monetized Machine Learning:

- Readily Available
- High Precision Results
- Model Security
- Data Security
- Sensitive Information Security

Model Extraction



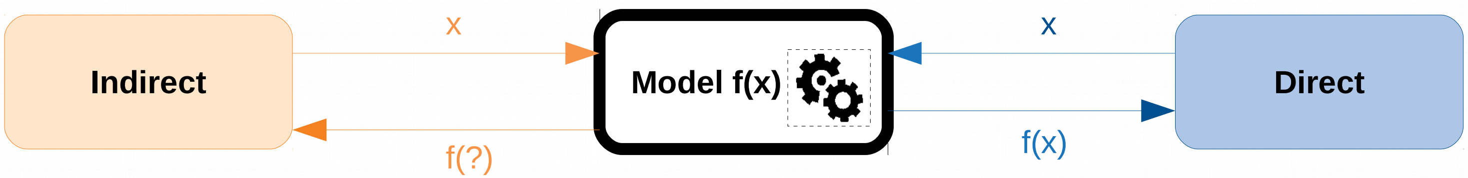
Goals of Model Extraction:

- Learn the close approximation of f using probing attacks to gain information.
- Get f'(x) = f(x)on ≥ 99.9% of inputs using as few queries as possible.

Threat Model

There are two adversarial models in practice

- **Direct queries:** adversary attack provides an arbitrary input x to a model f and obtains the output f(x)
- **Indirect queries:** adversary attack makes only indirect queries on points in input space M and yields outputs f(ex(M)). (ex is the extraction mechanism that may be unknown to the adversary)



Model Function

Assumptions and Parameters:

A model is a function f: X→Y. An input is a d-dimensional vector in the feature space X=X1×X2×⋯×Xd. Outputs lie in the range Y.

Test error Rtest: This is the average error over a test set D, given by Rtest(f,ˆf)=∑(x,y)∈Dd(f(x),ˆf(x))/|D|. **A low test error implies that f' matches f well for in-puts distributed like the training data samples.**

Uniform error Runif: For a set U of vectors uniformly chosen in X, let Runif(f,ˆf)=∑x∈Ud(f(x),ˆf(x))/|U|.Thus **Runif estimates the fraction of the full feature space on which f and f' disagree.**

Applications (Attack Scenarios)

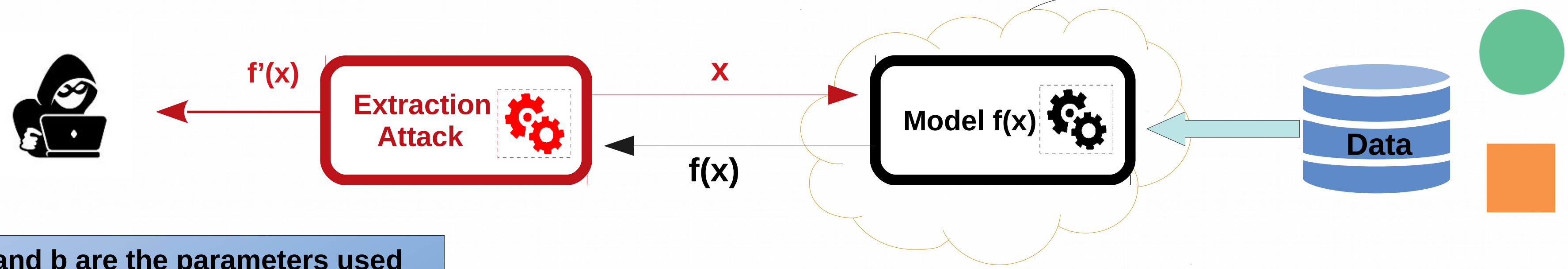
Why steal a machine learning model?

1. Undermine pay-for-prediction pricing model
2. Facilitate privacy attacks
3. Stepping stone to model-evasion [Lowd, Meek –2005] [Srndic, Laskov–2014]

Logistic Regression

Binary Classifier: Circle or Square?

$$f(x) = \frac{1}{1+e^{(-(w*x+b))}}$$



w and b are the parameters used by the training set to minimize error.

f maps features to predicted probability of the object being a circle (>0.5)

The Attack?

The extraction attack makes random N+1 queries to the model and receives f(x)

The attacker then solves for w and b using a system of equations:

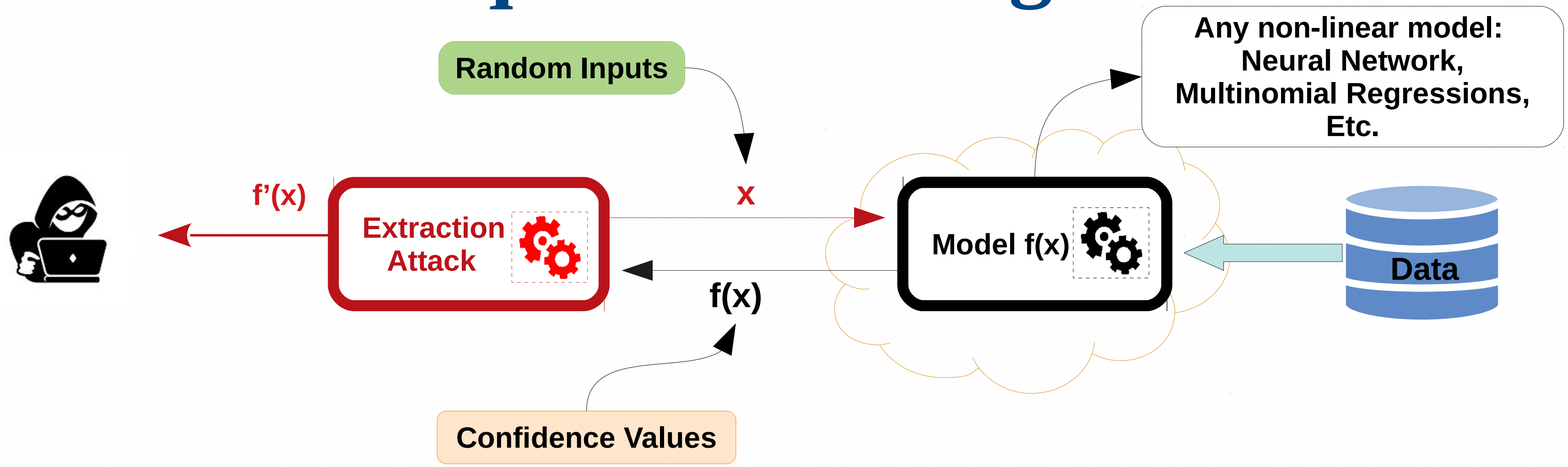
$$\ln\left(\frac{f(x)}{1-f(x)}\right) = w*x+b$$

Multiclass LRs and Multilayer Perceptrons

Using the same concept, this model extraction technique can be applied to more complex and multi-class models that utilize logistic regression. Below are the results of applying the extraction attack to models trained on the Adult data set with multiclass target ‘Race’.

Model	Unknowns	Queries	$1 - R_{test}$	$1 - R_{unif}$	Time (s)
Softmax	530	265	99.96%	99.75%	2.6
		530	100.00%	100.00%	3.1
OvR	530	265	99.98%	99.98%	2.8
		530	100.00%	100.00%	3.5
MLP	2,225	1,112	98.17%	94.32%	155
		2,225	98.68%	97.23%	168
		4,450	99.89%	99.82%	195
		11,125	99.96%	99.99%	89

Non-Linear Equation Solving Attacks



Solve the system of non-linear equations: It becomes a noiseless optimization problem with gradient descent.

The authors were able to achieve ~99.9% Accuracy using this method with Multinomial Regressions and Deep Neural Networks.

Decision Trees

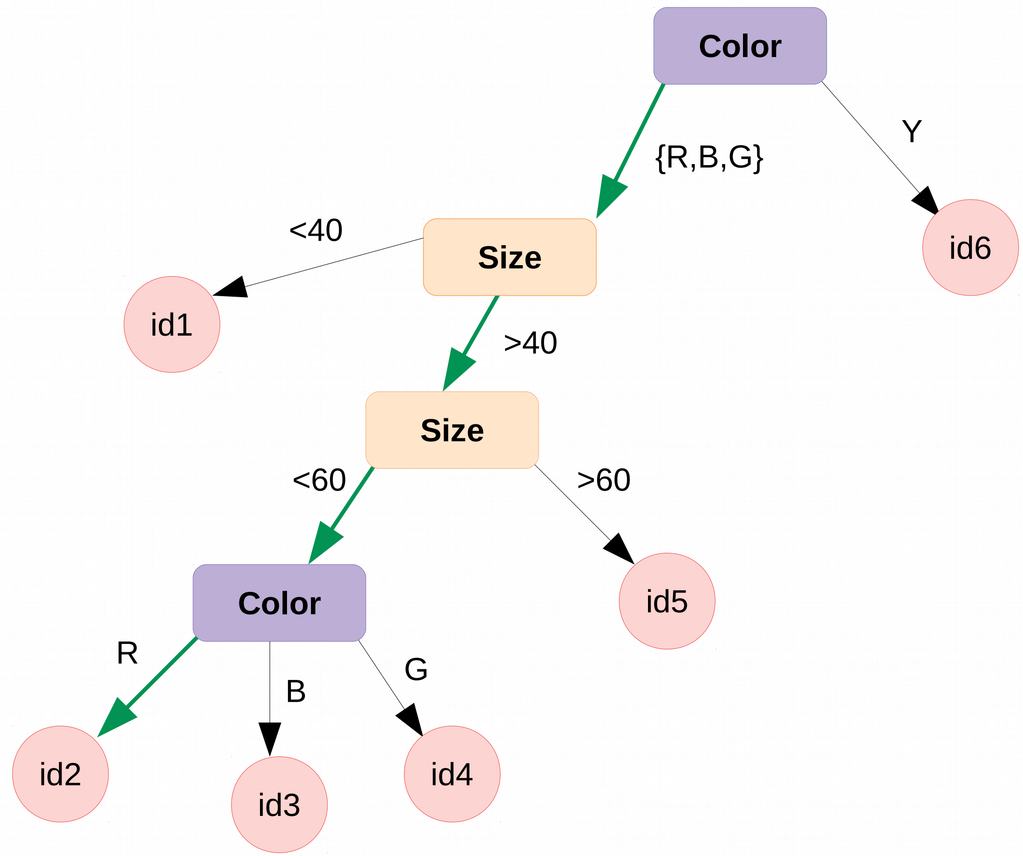
Decision trees do not compute class probabilities as a continuous function of their input.

Decision trees partition the input space into discrete regions, each of which is assigned a label and confidence score.

Path-finding attack (Algorithm) that assumes a leaf-identity oracle that returns unique identifiers for each leaf.

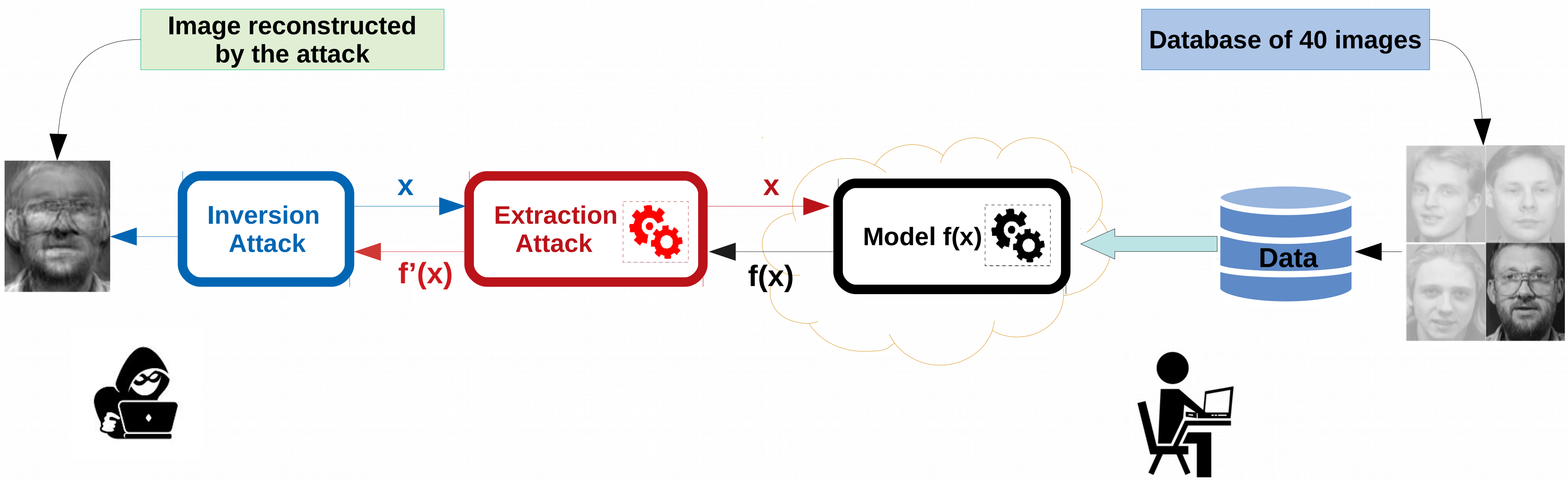
Algorithm searches for and finds each leaf (id) and path in the decision tree.

Decision Tree for Color and Size Features



Model Inversion Attacks on Extracted Models

By adding an additional inversion attack, an adversary could potentially steal not only the model, but also the sensitive data that was used to train it.



Non-Equation Solving Attack

- Extend the Lowd-Meek approach to non-linear models

- Active Learning: Query points close to “decision boundary”-Update f’ to fit these points

- Multinomial Regressions, Neural Networks, SVMs:->99% agreement between f and f’-≈ 100 queries per model parameter of f

Machine Learning as a Service

Focus: extracting models set up by random users, who wish to charge for predictions

Focus: extracting a model trained by the authors themselves, but to which they only have black-box access

Big ML Results

Model	Leaves	Unique IDs	Depth	Without incomplete queries $1-R_{test}$	Without incomplete queries $1-R_{unif}$	Without incomplete queries Queries	With incomplete queries $1-R_{test}$	With incomplete queries $1-R_{unif}$	With incomplete queries Queries
IRS Tax Patterns	318	318	8	100.00%	100.00%	101,057	100.00%	100.00%	29,609
Steak Survey	193	28	17	92.45%	86.40%	3,652	100.00%	100.00%	4,013
GSS Survey	159	113	8	99.98%	99.61%	7,434	100.00%	99.65%	2,752
Email Importance	109	55	17	99.13%	99.90%	12,888	99.81%	99.99%	4,081
Email Spam	219	78	29	87.20%	100.00%	42,324	99.70%	100.00%	21,808
German Credit	26	25	11	100.00%	100.00%	1,722	100.00%	100.00%	1,150
Medical Cover	49	49	11	100.00%	100.00%	5,966	100.00%	100.00%	1,788
Bitcoin Price	155	155	9	100.00%	100.00%	31,956	100.00%	100.00%	7,390

Table 6: **Performance of extraction attacks on public models from BigML.** For each model, we report the number of leaves in the tree, the number of unique identifiers for those leaves, and the maximal tree depth. The chosen granularity ε for continuous features is 10^{-3}.

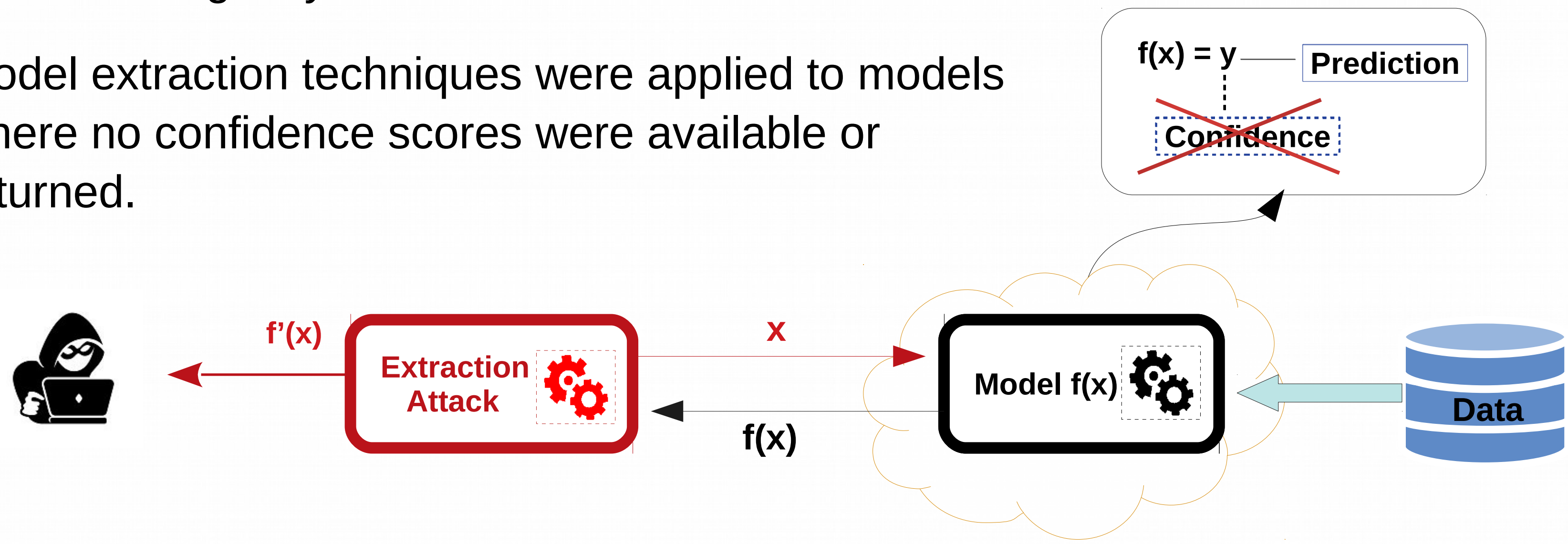
Amazon AWS Results

Model	OHE	Binning	Queries	Time (s)	Price ($)
Circles	-	Yes	278	28	0.03
Digits	-	No	650	70	0.07
Iris	-	Yes	644	68	0.07
Adult	Yes	Yes	1,485	149	0.15

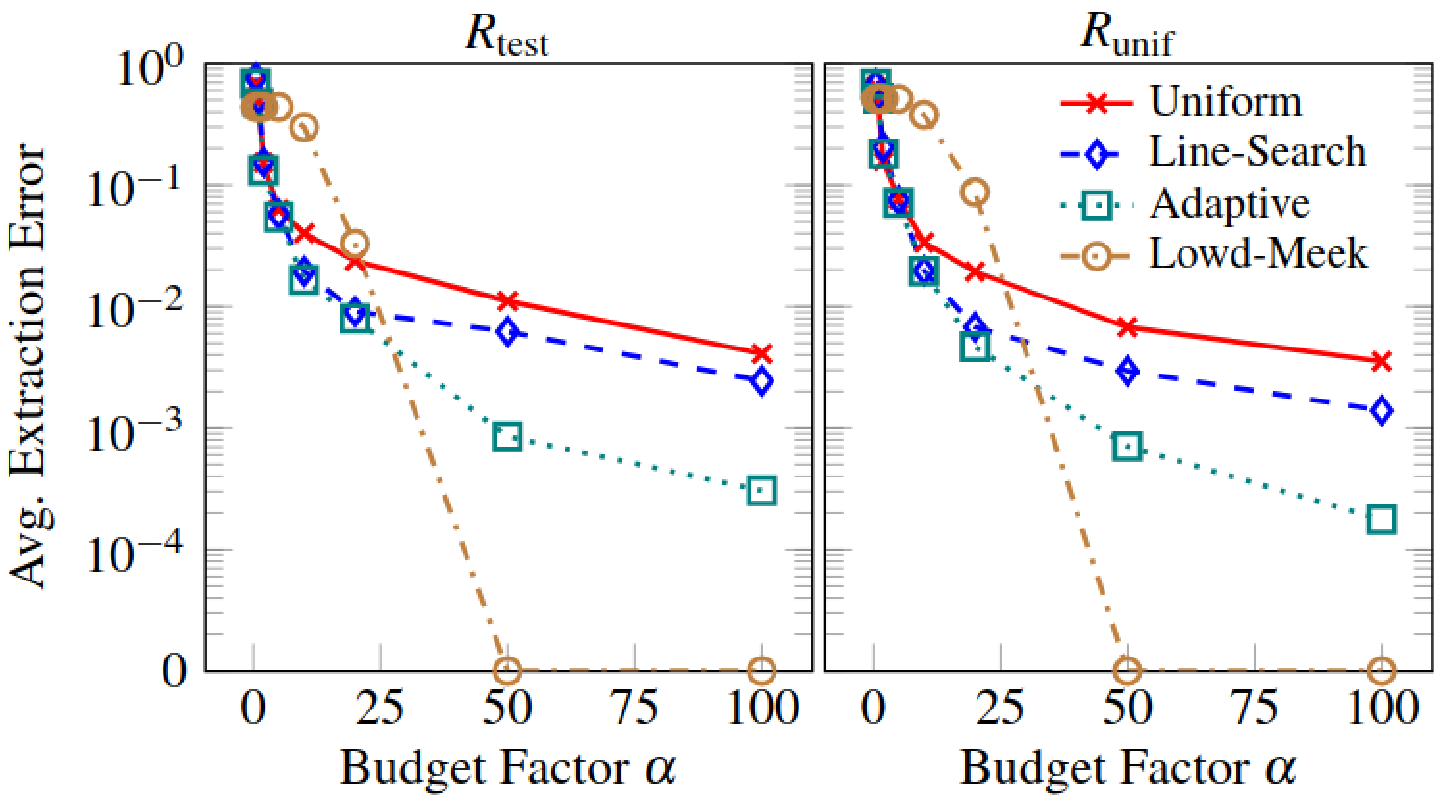
Table 7: **Results of model extraction attacks on Amazon.** OHE stands for one-hot-encoding. The reported query count is the number used to find quantile bins (at a granularity of 10^{-3}), plus those queries used for equation-solving. Amazon charges $0.0001 per prediction [1].

Limited Information

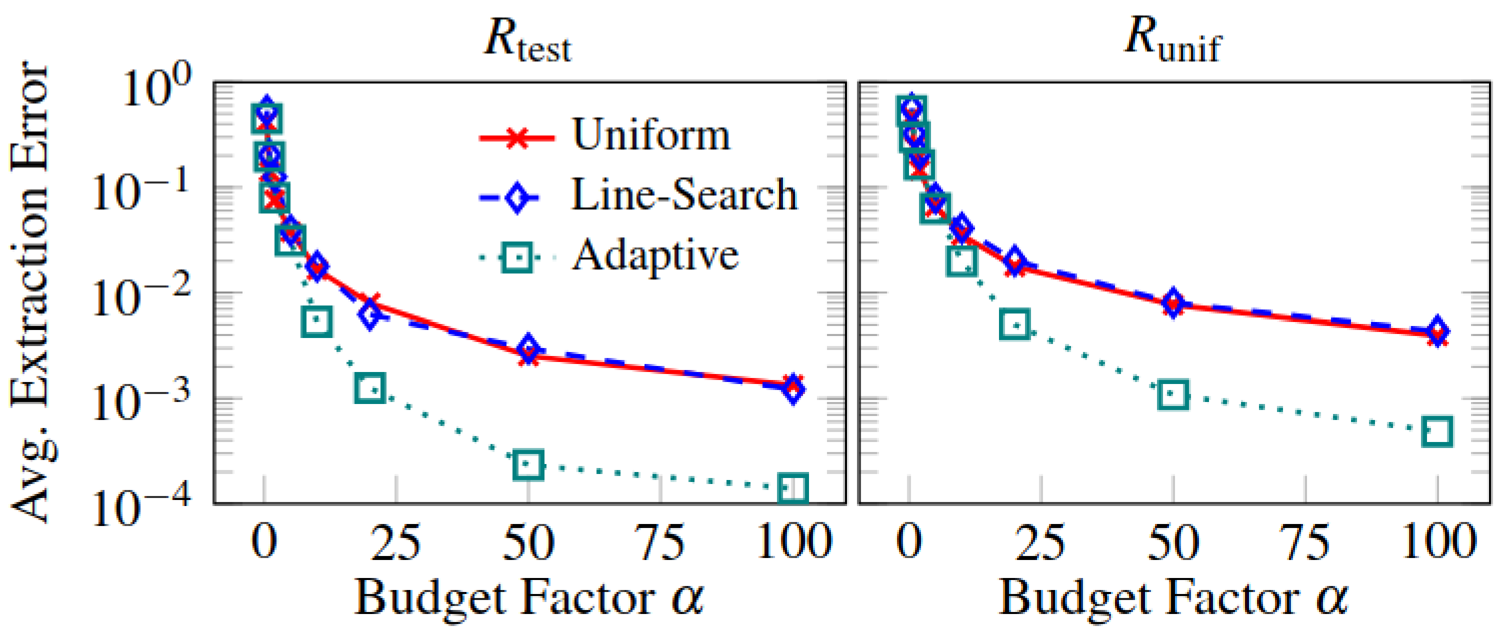
- The results of the authors experiment suggested that returning only labels would be safer.
- Model extraction techniques were applied to models where no confidence scores were available or returned.



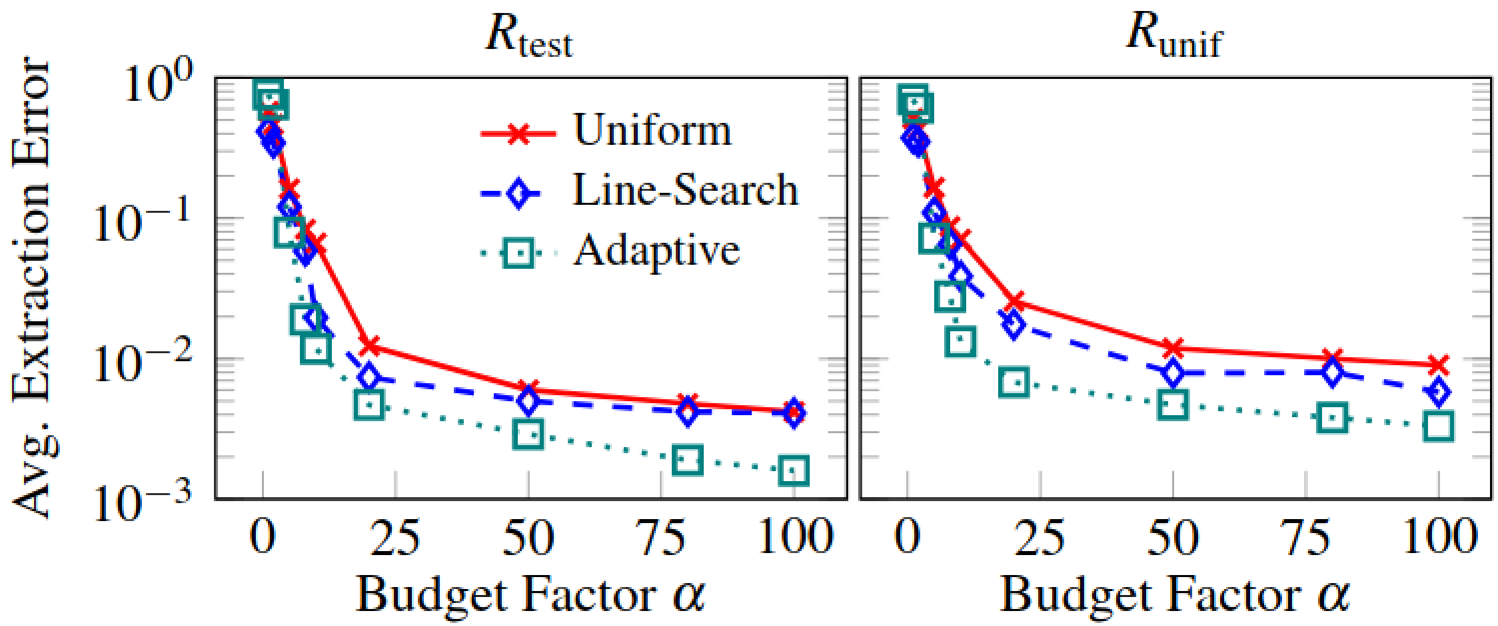
Linear Models



Softmax Models



Support Vector Machines (SVMs) with Radial Basis Function (RBF) Kernels



Code Demonstration

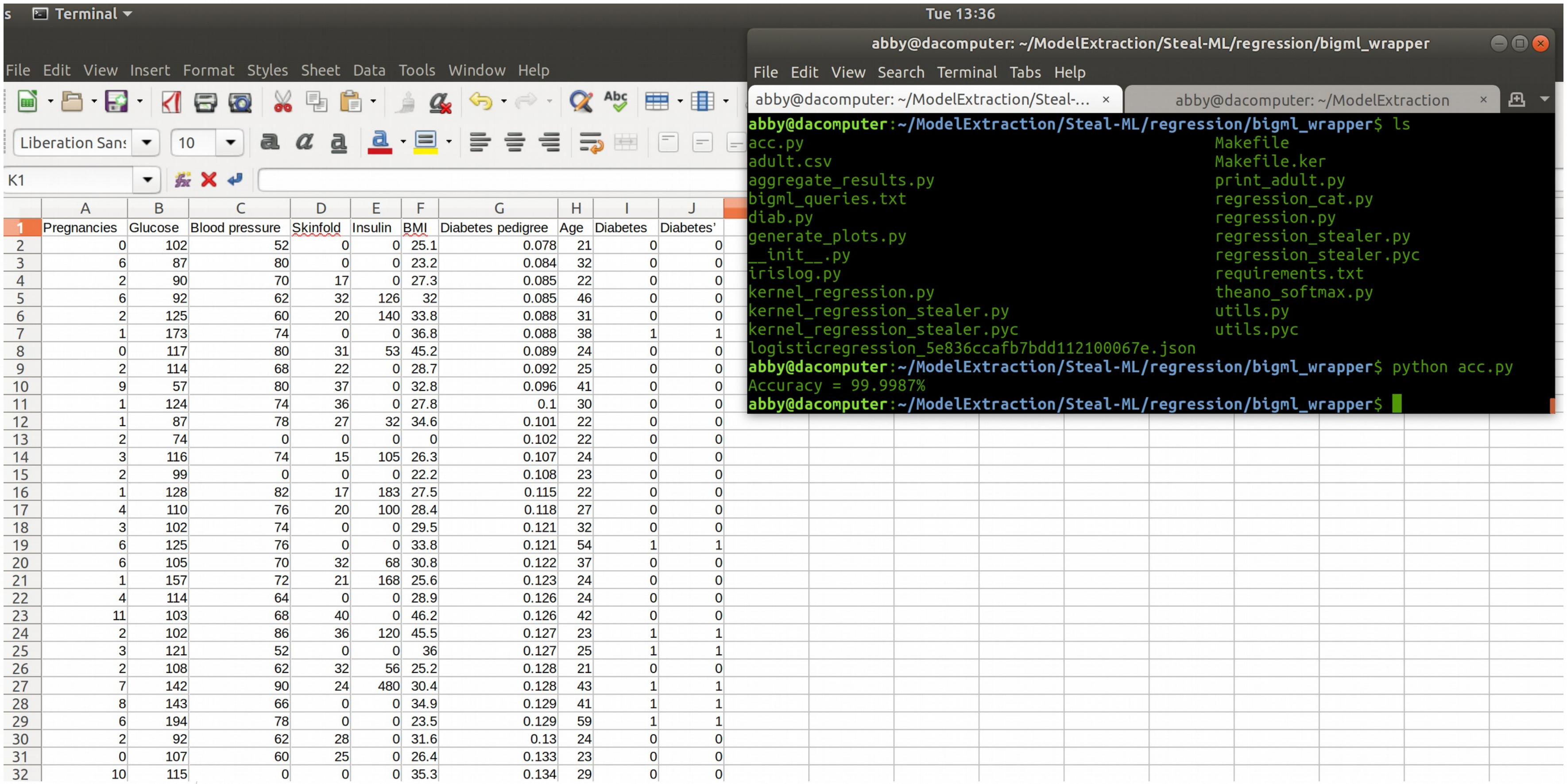
Code Issues:

- Poor Documentation
- Missing Data/Models
- Outdated Libraries

Code Demonstration


```
obtained train accuracy of 1.0
opti ran for 0.02 s
iris,passive,300,extr,0.00e+00,0.00e+00,2.92e-09,5.15e-09,5.60e-07
iris,passive,300,base,1.33e-02,3.07e-02,7.35e-01,7.33e-01,7.55e+01
finding solution of system of 750 equations with 15 unknowns with BFGS
Optimization terminated successfully.
         Current function value: 546.374088
         Iterations: 37
         Function evaluations: 43
         Gradient evaluations: 43
obtained train accuracy of 1.0
opti ran for 0.04 s
iris,passive,750,extr,0.00e+00,0.00e+00,4.91e-10,1.10e-09,8.01e-08
iris,passive,750,base,3.33e-02,3.52e-02,7.38e-01,7.27e-01,7.64e+01
finding solution of system of 1500 equations with 15 unknowns with BFGS
Optimization terminated successfully.
         Current function value: 1095.086908
         Iterations: 38
         Function evaluations: 44
         Gradient evaluations: 44
obtained train accuracy of 1.0
opti ran for 0.07 s
iris,passive,1500,extr,0.00e+00,0.00e+00,1.43e-09,1.48e-09,1.15e-07
iris,passive,1500,base,6.67e-03,2.85e-02,7.35e-01,7.29e-01,7.36e+01
```

Code Demonstration



	A	B	C	D	E	F	G	H	I	J
1	Pregnancies	Glucose	Blood pressure	Skinfold	Insulin	BMI	Diabetes pedigree	Age	Diabetes	Diabetes'
2	0	102	52	0	0	25.1	0.078	21	0	0
3	6	87	80	0	0	23.2	0.084	32	0	0
4	2	90	70	17	0	27.3	0.085	22	0	0
5	6	92	62	32	126	32	0.085	46	0	0
6	2	125	60	20	140	33.8	0.088	31	0	0
7	1	173	74	0	0	36.8	0.088	38	1	1
8	0	117	80	31	53	45.2	0.089	24	0	0
9	2	114	68	22	0	28.7	0.092	25	0	0
10	9	57	80	37	0	32.8	0.096	41	0	0
11	1	124	74	36	0	27.8	0.1	30	0	0
12	1	87	78	27	32	34.6	0.101	22	0	0
13	2	74	0	0	0	0	0.102	22	0	0
14	3	116	74	15	105	26.3	0.107	24	0	0
15	2	99	0	0	0	22.2	0.108	23	0	0
16	1	128	82	17	183	27.5	0.115	22	0	0
17	4	110	76	20	100	28.4	0.118	27	0	0
18	3	102	74	0	0	29.5	0.121	32	0	0
19	6	125	76	0	0	33.8	0.121	54	1	1
20	6	105	70	32	68	30.8	0.122	37	0	0
21	1	157	72	21	168	25.6	0.123	24	0	0
22	4	114	64	0	0	28.9	0.126	24	0	0
23	11	103	68	40	0	46.2	0.126	42	0	0
24	2	102	86	36	120	45.5	0.127	23	1	1
25	3	121	52	0	0	36	0.127	25	1	1
26	2	108	62	32	56	25.2	0.128	21	0	0
27	7	142	90	24	480	30.4	0.128	43	1	1
28	8	143	66	0	0	34.9	0.129	41	1	1
29	6	194	78	0	0	23.5	0.129	59	1	1
30	2	92	62	28	0	31.6	0.13	24	0	0
31	0	107	60	25	0	26.4	0.133	23	0	0
32	10	115	0	0	0	35.3	0.134	29	0	0

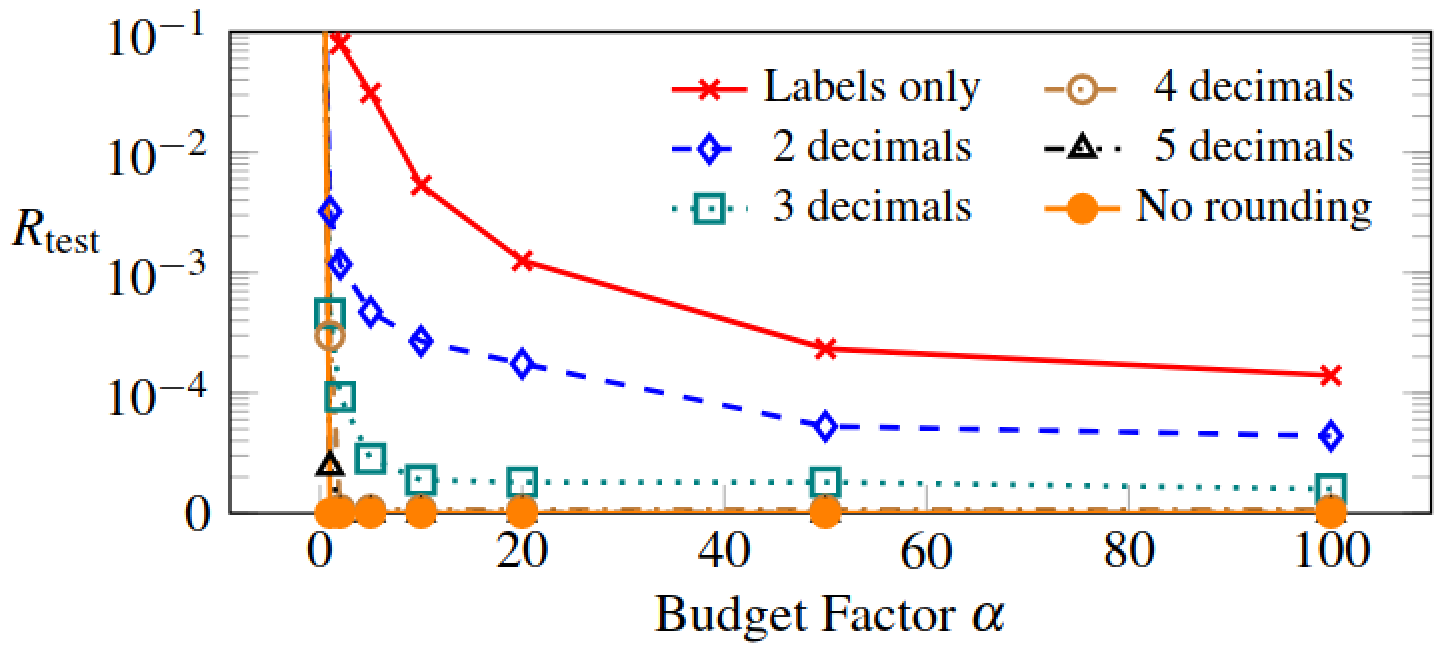
```
abby@dacomputer:~/ModelExtraction/Steal-ML/regression/bigml_wrapper$ ls
acc.py                                              Makefile
adult.csv                                           Makefile.ker
aggregate_results.py                                print_adult.py
bigml_queries.txt                                   regression_cat.py
diab.py                                             regression.py
generate_plots.py                                   regression_stealer.py
__init__.py                                         regression_stealer.pyc
irislog.py                                          requirements.txt
kernel_regression.py                                theano_softmax.py
kernel_regression_stealer.py                        utils.py
kernel_regression_stealer.pyc                       utils.pyc
logisticregression_5e836ccafb7bdd112100067e.json
abby@dacomputer:~/ModelExtraction/Steal-ML/regression/bigml_wrapper$ python acc.py
Accuracy = 99.9987%
abby@dacomputer:~/ModelExtraction/Steal-ML/regression/bigml_wrapper$
```

Performance Analysis Overview

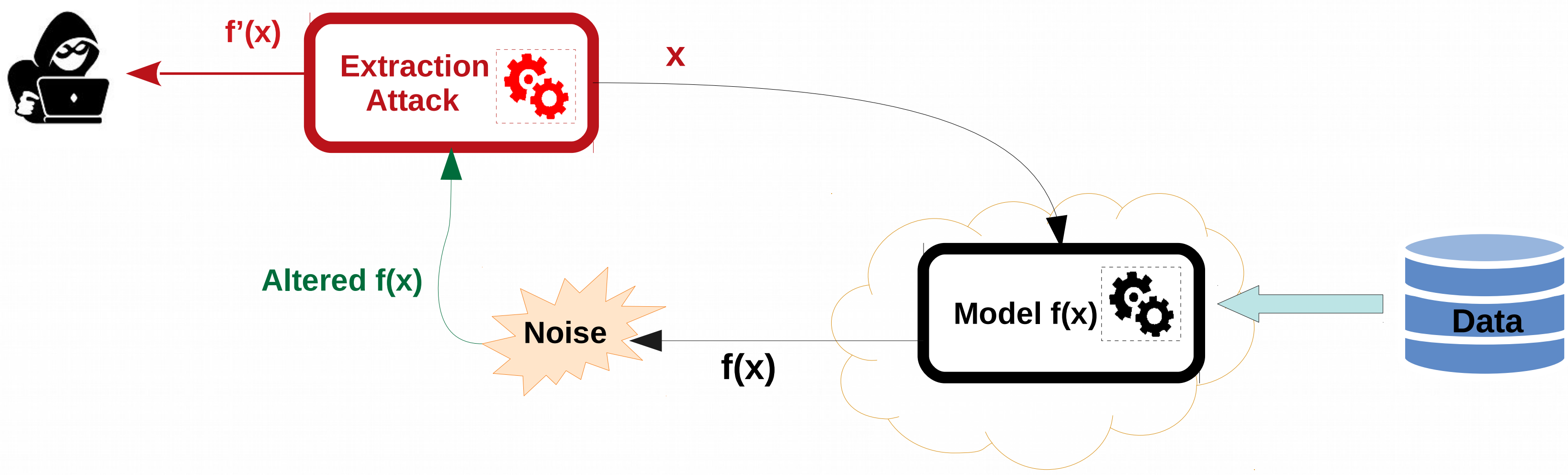
With the use of various methods, authors were able to successfully extract models that produced over 99% accuracy.

Rounding Confidences

Rounding the confidence values will cause more error when extracting the models, however including only labels proves the most effective.



Differential Privacy



Conclusion

The authors demonstrated how the flexible prediction APIs exposed by current Machine Learning as a Service providers enable new model extraction attacks that could subvert model monetization, violate training-data privacy, and facilitate model evasion.

Questions?