Chapter 6: Understanding Machine Learning-Based Malware Detectors

DATA SCIENCE IN SECURITY

Introduction

- With the open source machine learning tools available today
  - you can build custom malware detection tools
    - whether as your primary detection tool
    - or to supplement commercial solutions.

Introduction

- why build your own machine learning tools?
  - can allow you to catch new examples of threats
    - commercial antivirus engines might miss them
  - Commercial tools are “closed books”
    - we don’t necessarily know how they work
    - we have limited ability to tune them
  - we know how our own detection tools work
    - can tune them to our liking to reduce
      - false positives
      - or false negatives

Steps for Building a Machine learning–Based Detector

- fundamental difference between machine learning and other kinds of computer algorithms.
  - traditional algorithms tell the computer what to do
  - machine-learning systems learn how to solve a problem by example
    - they automate the work of creating security signatures
    - they have the potential to perform more accurately than signature-based approaches
      - especially on new, previously unseen malware.

Steps for Building a Machine learning–Based Detector

- workflow to build machine learning-based detector
  1. Collect examples of malware and benignware.
     - called training examples to train the machine learning system to recognize malware.
  2. Extract features from each training example
     - to represent the example as an array of numbers
     - also includes research to design good features
  3. Train the machine learning system
     - using the features we have extracted.
  4. Test the approach on some data
     - not included in our training examples

Gathering Training Examples

- ability to recognize suspicious binaries depends heavily on the
  - quantity of training examples
    - the more examples you feed your system, the more accurate it's likely to be
  - quality of training examples you provide.
    - samples you collect should mirror the kind of malware and benignware you expect your detector to see

Extracting Features

- we train machine learning systems by showing them features of software binaries
  - file attributes that will help the system distinguish between good and bad files e.g.
    - Whether it’s digitally signed
    - The presence of malformed headers
    - The presence of encrypted data
    - Whether it has been seen on more than 100 network workstations

Designing Good Features

- choose features that represent your best guess to distinguish bad files from good files.
  - E.g. “contains encrypted data”
    - benignware will contain encrypted data more rarely.
- don’t use set of features too large relative to the number of training examples
  - not enough training examples to teach your system what each feature actually says (probably)
  - Statistics tells us that it’s better to
    - give your system a few features relative to the training examples
    - Let it form well-founded beliefs about which features truly indicate malware.

Designing Good Features

- make sure features represent a range of hypotheses
  - E.g.
    - you may choose features related to encryption
    - but make sure to also use features unrelated to encryption
      - if system fails to detect malware based on one type of feature, it might still detect it using other features.

Training Machine Learning Systems

- depends on the machine learning approach you're using
  - E.g.
    - training a decision tree approach involves a different learning algorithm than training a logistic regression approach
- Fortunately
  - all machine learning detectors provide the same basic interface
    - You provide them with training data that contains
      - features from sample binaries
      - corresponding labels
    - the algorithms learn to determine
      - a new unseen binaries are malicious or benign.

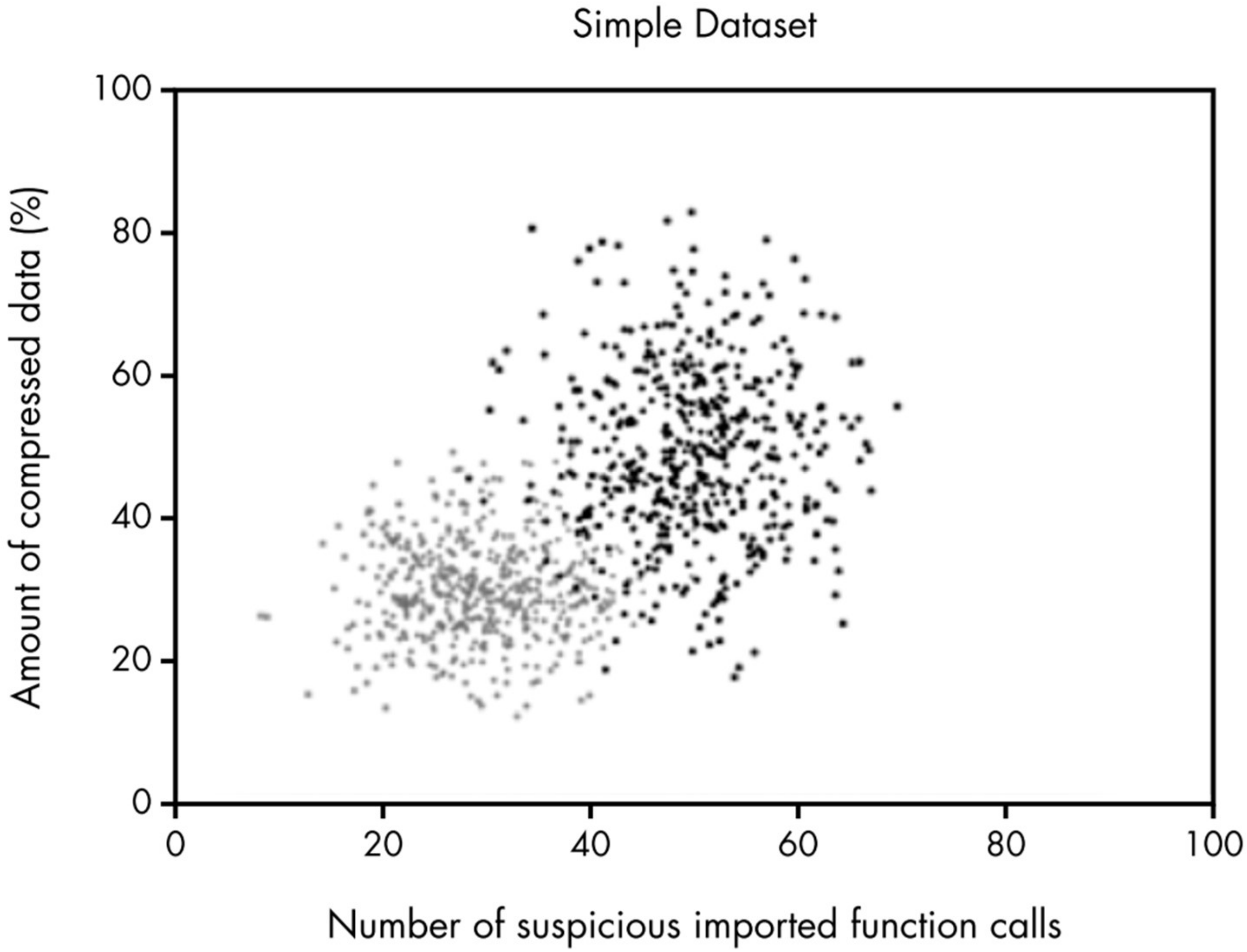
Testing Machine Learning Systems

- you need to check how accurate it is.
  - running the trained system on data that you didn’t train it on and seeing
    - how well our systems will detect new malware
    - how well our systems will avoid producing false positives on new benignware.

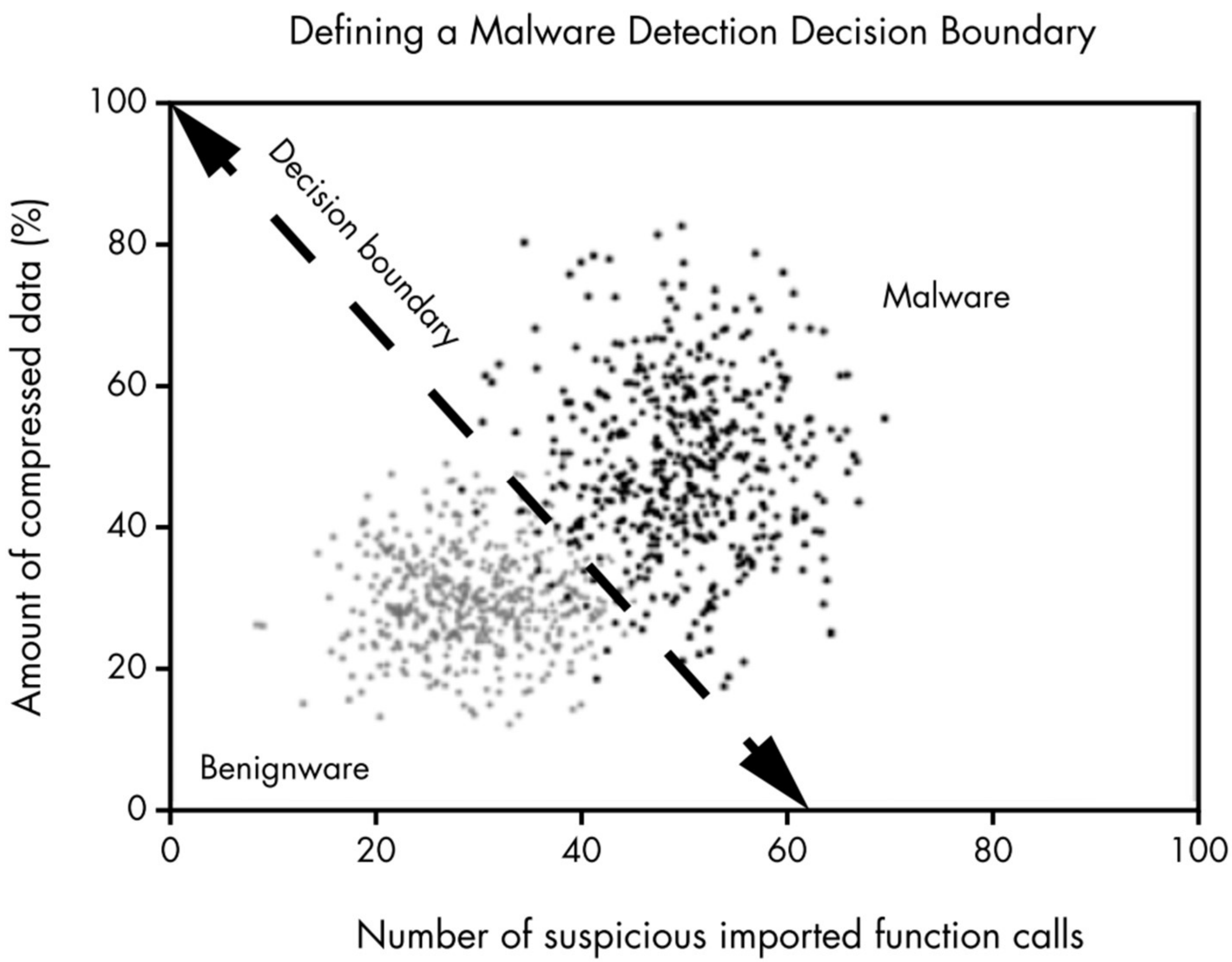
understanding Feature Spaces and Decision Boundaries

- Two simple geometric ideas can help you understand all machine learning-based detection algorithms:
  - the idea of a geometrical feature space
    - geometrical space defined by the features you've selected
  - the idea of a decision boundary.
    - geometrical structure running through feature space such that
      - binaries on one side are defined as malware
      - binaries on the other side are defined as benignwareSS

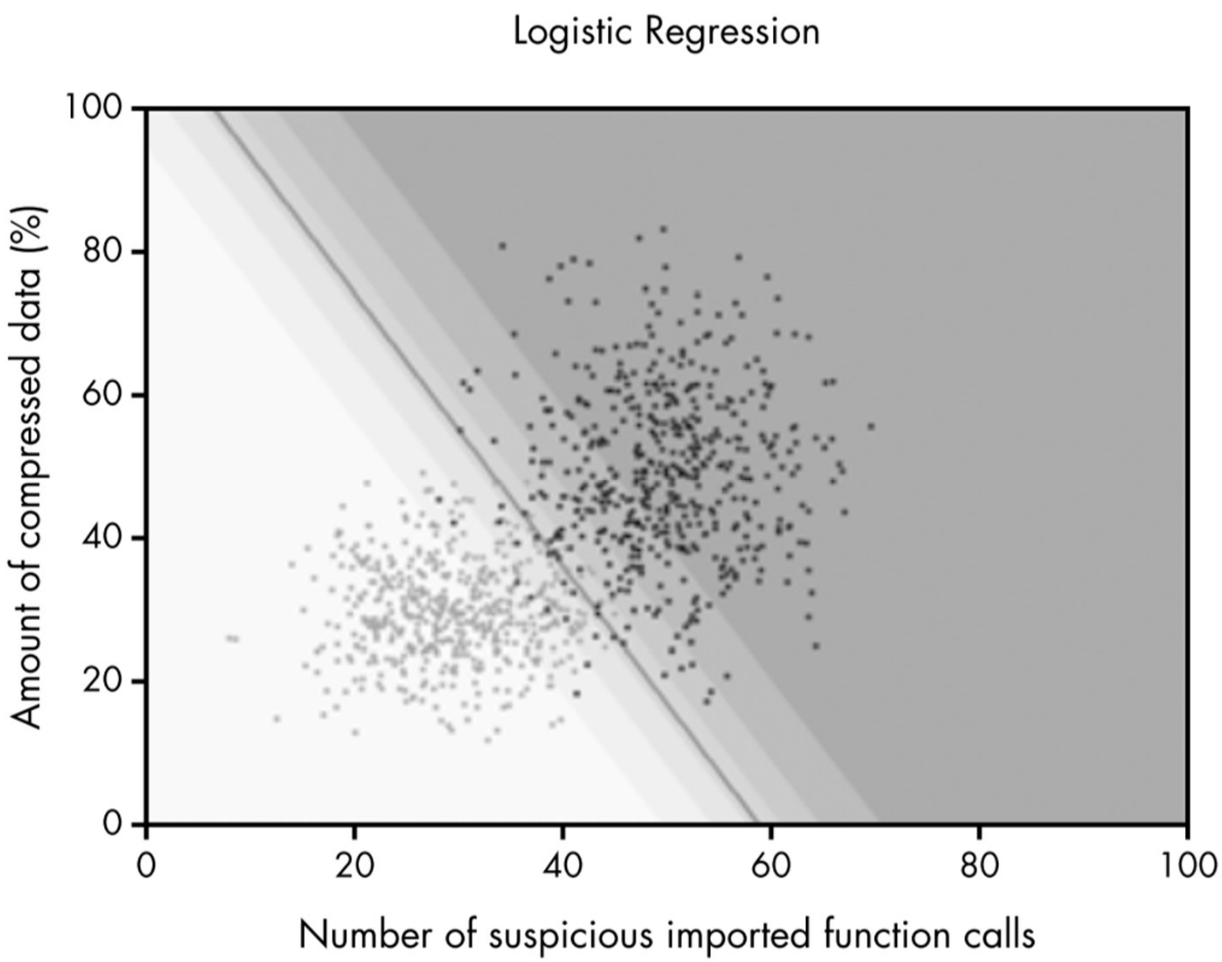
understanding Feature Spaces and Decision Boundaries



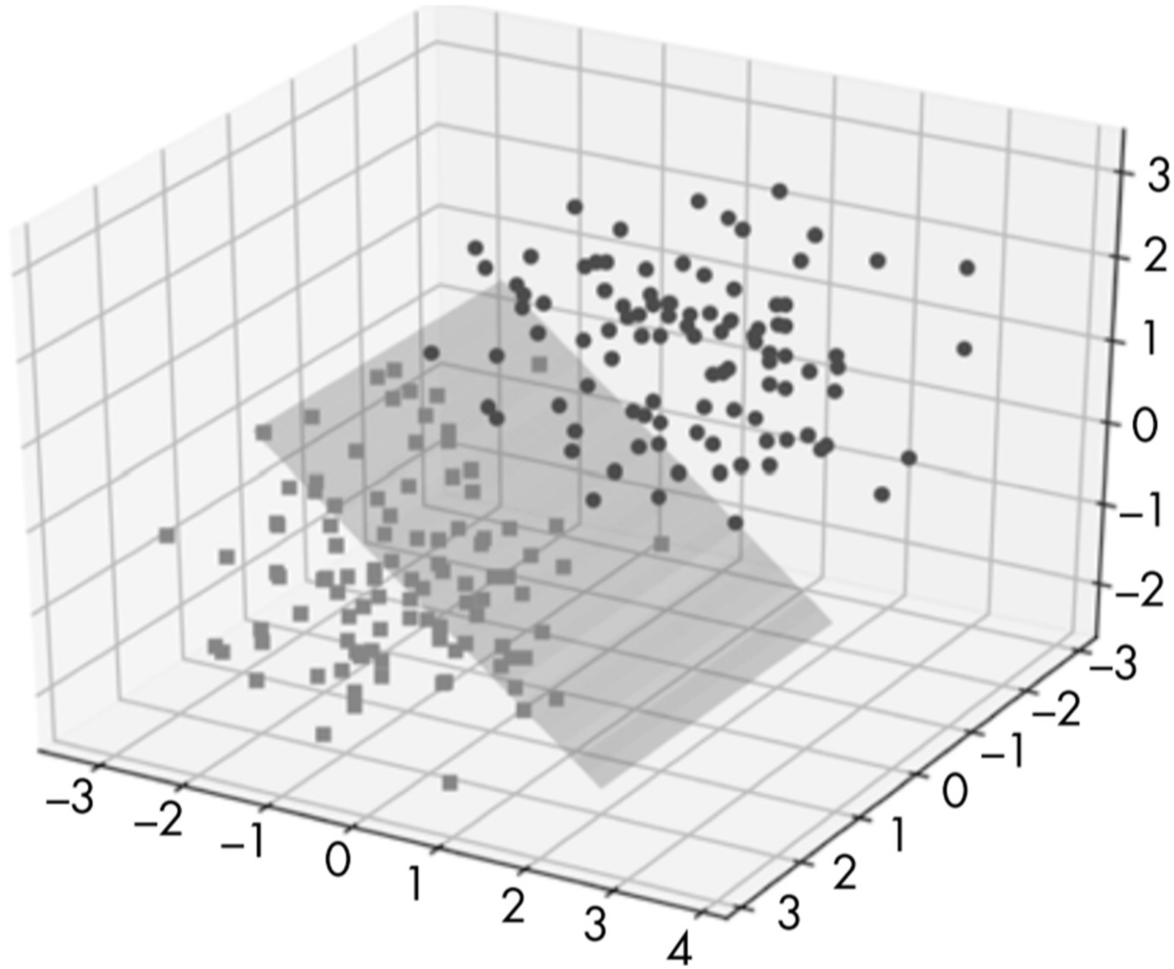
understanding Feature Spaces and Decision Boundaries



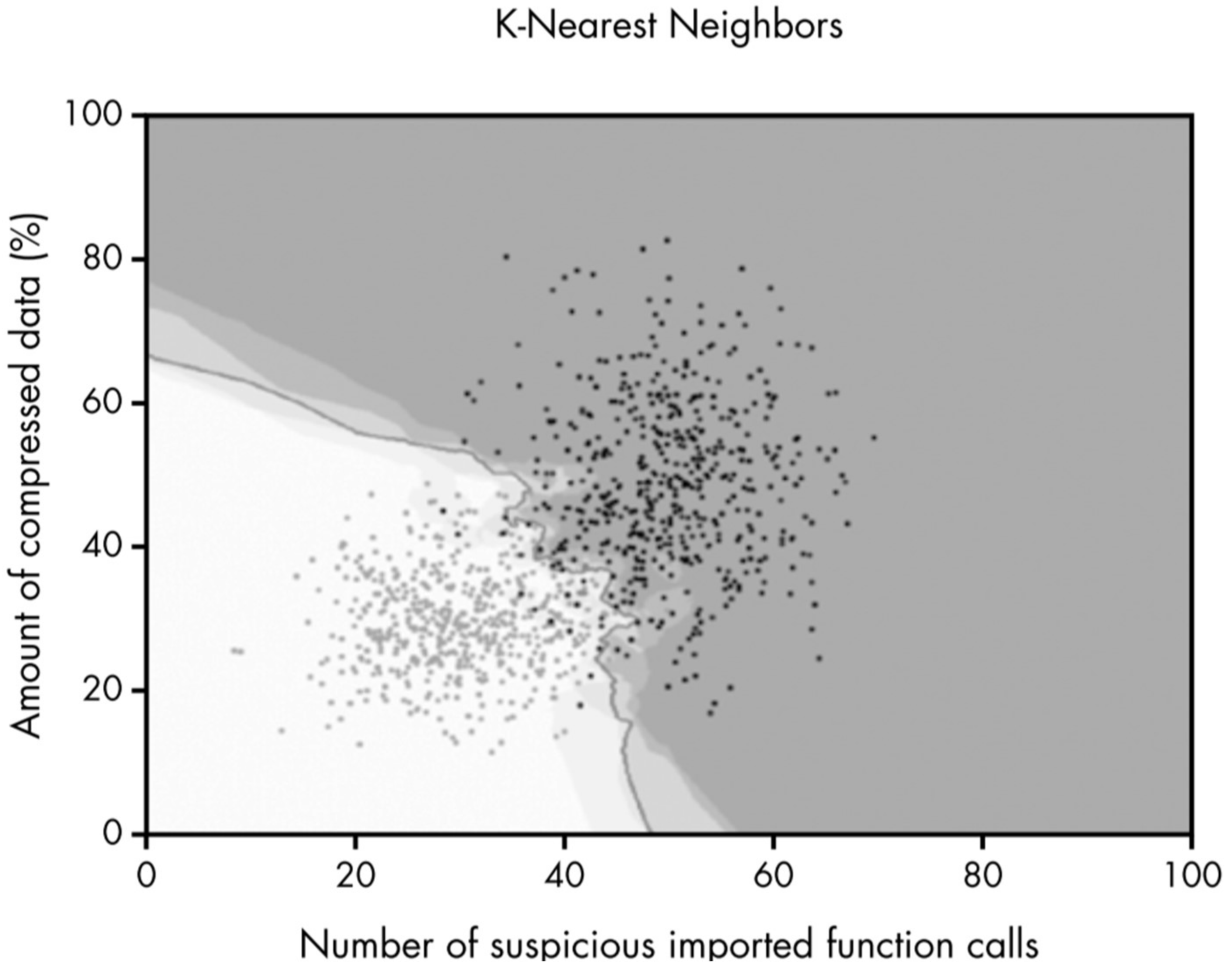
understanding Feature Spaces and Decision Boundaries



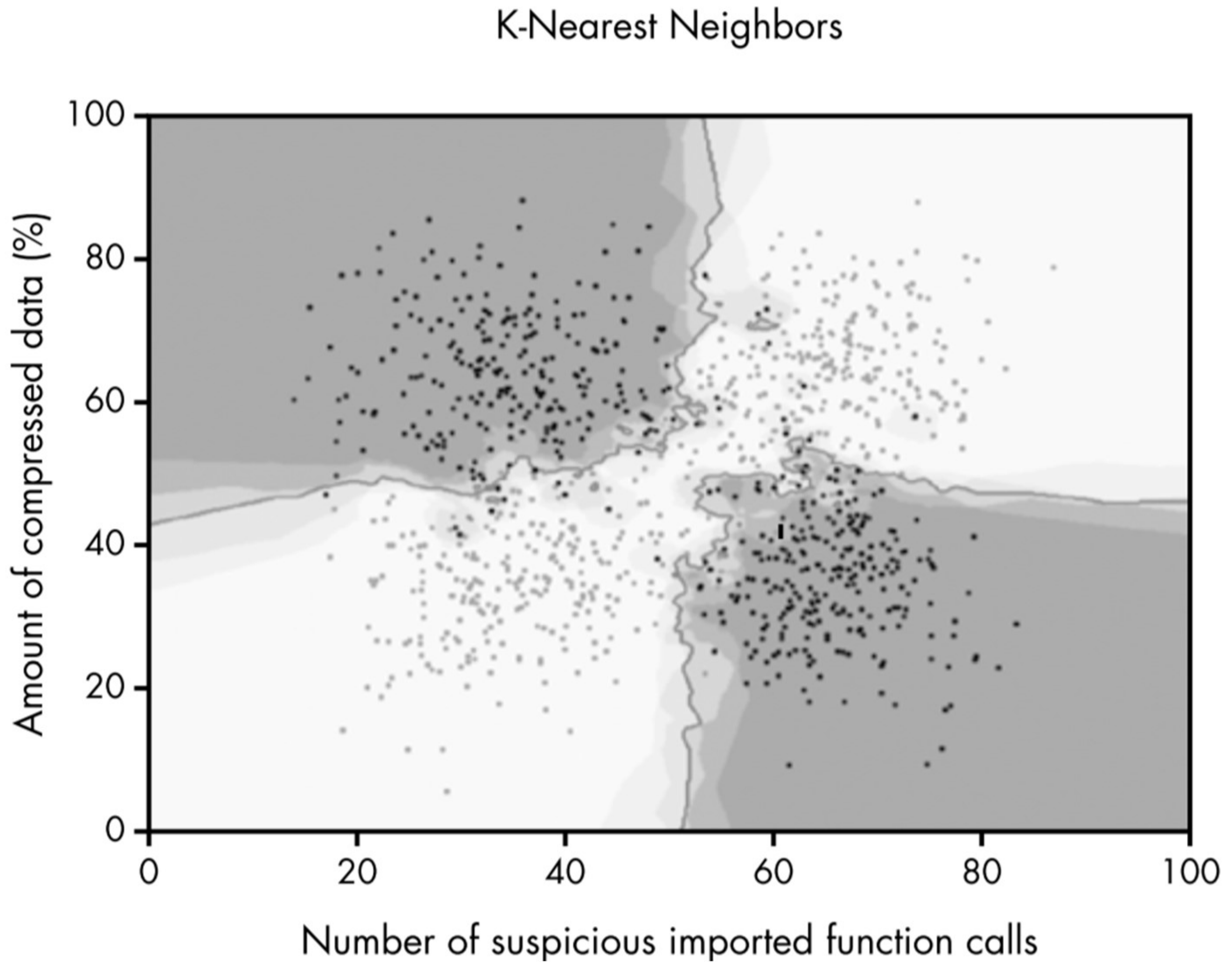
understanding Feature Spaces and Decision Boundaries



understanding Feature Spaces and Decision Boundaries



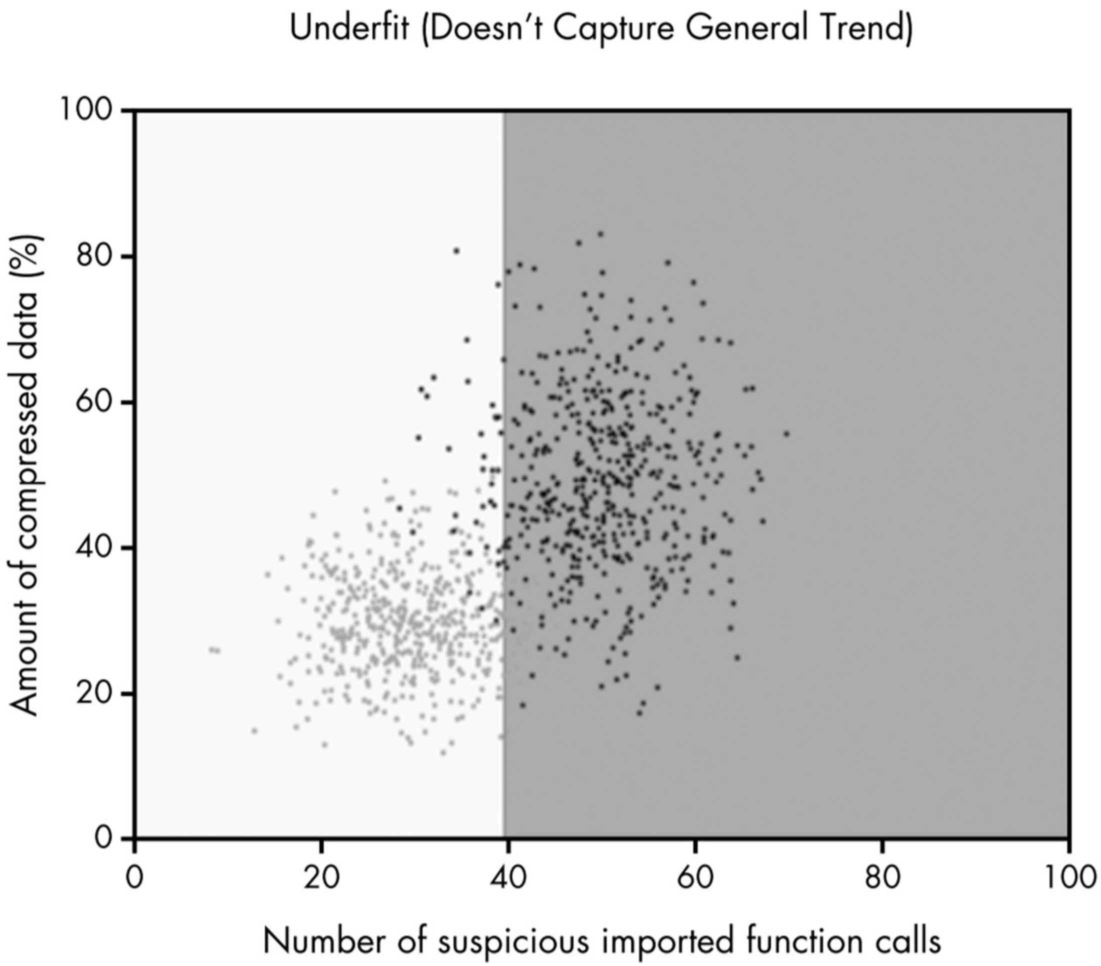
understanding Feature Spaces and Decision Boundaries



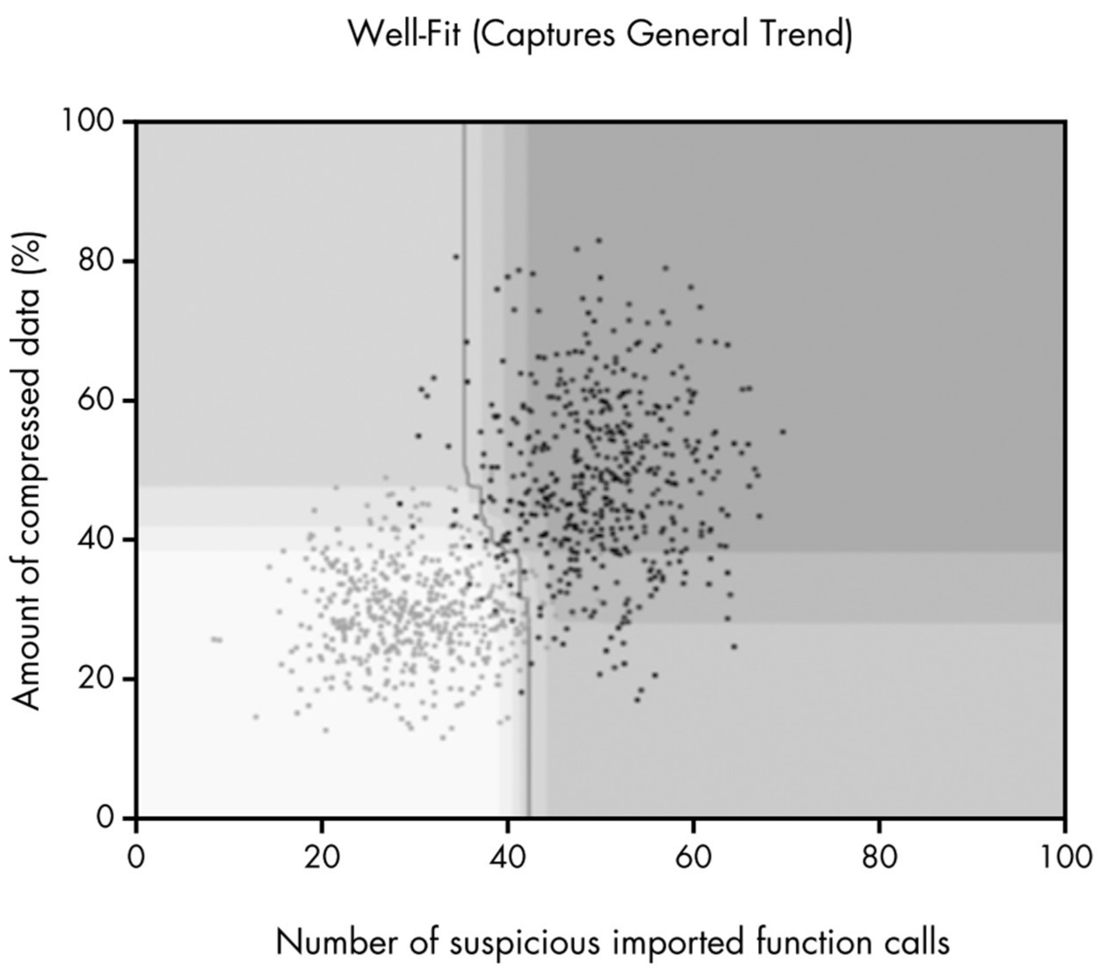
what Makes Models Good or Bad: overfitting and underfitting

- Good, accurate detection models
  - capture the general trend in what the training data says
  - without getting distracted by the outliers or the exceptions
- Underfit models
  - ignore outliers
  - but fail to capture the general trend
  - resulting in poor accuracy on unseen binaries
- Overfit models
  - get distracted by outliers
  - don’t reflect the general trend
  - yield poor accuracy on unseen binaries.

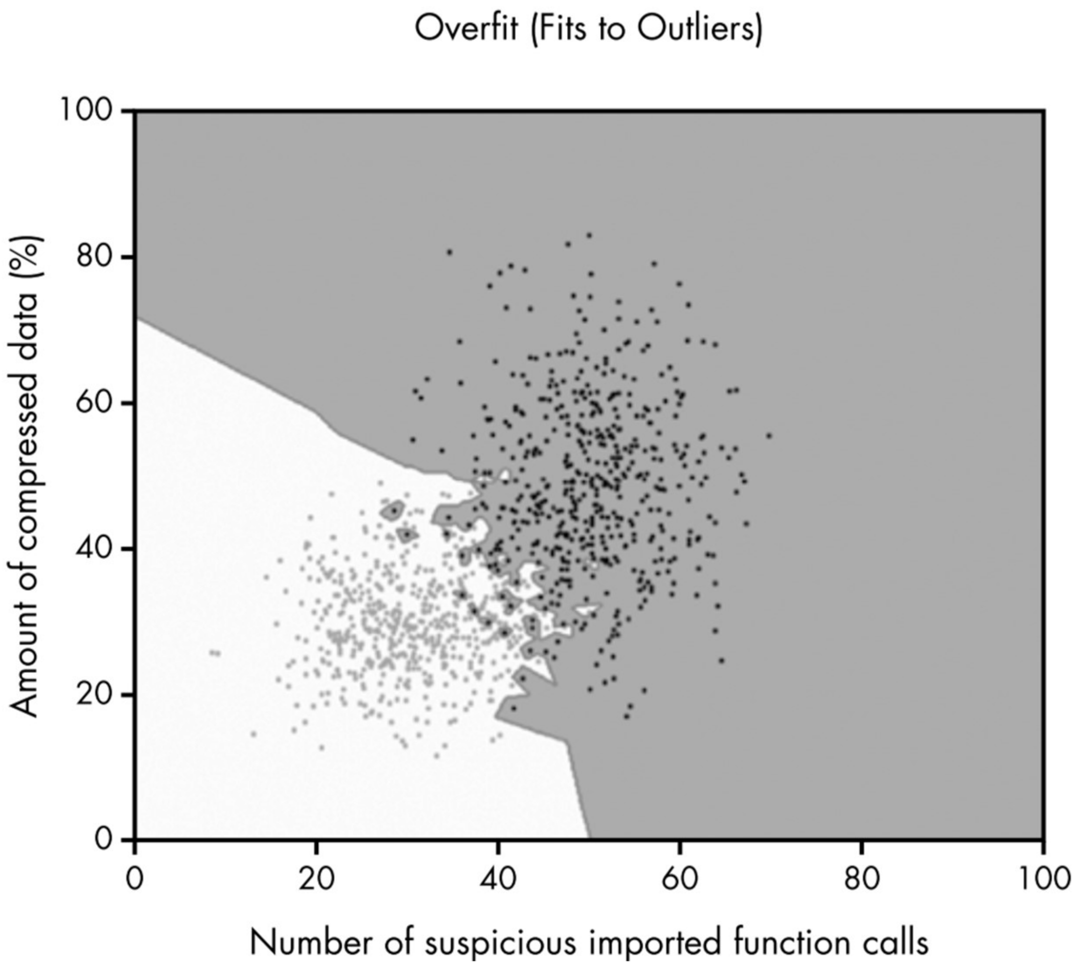
what Makes Models Good or Bad: overfitting and underfitting



what Makes Models Good or Bad: overfitting and underfitting

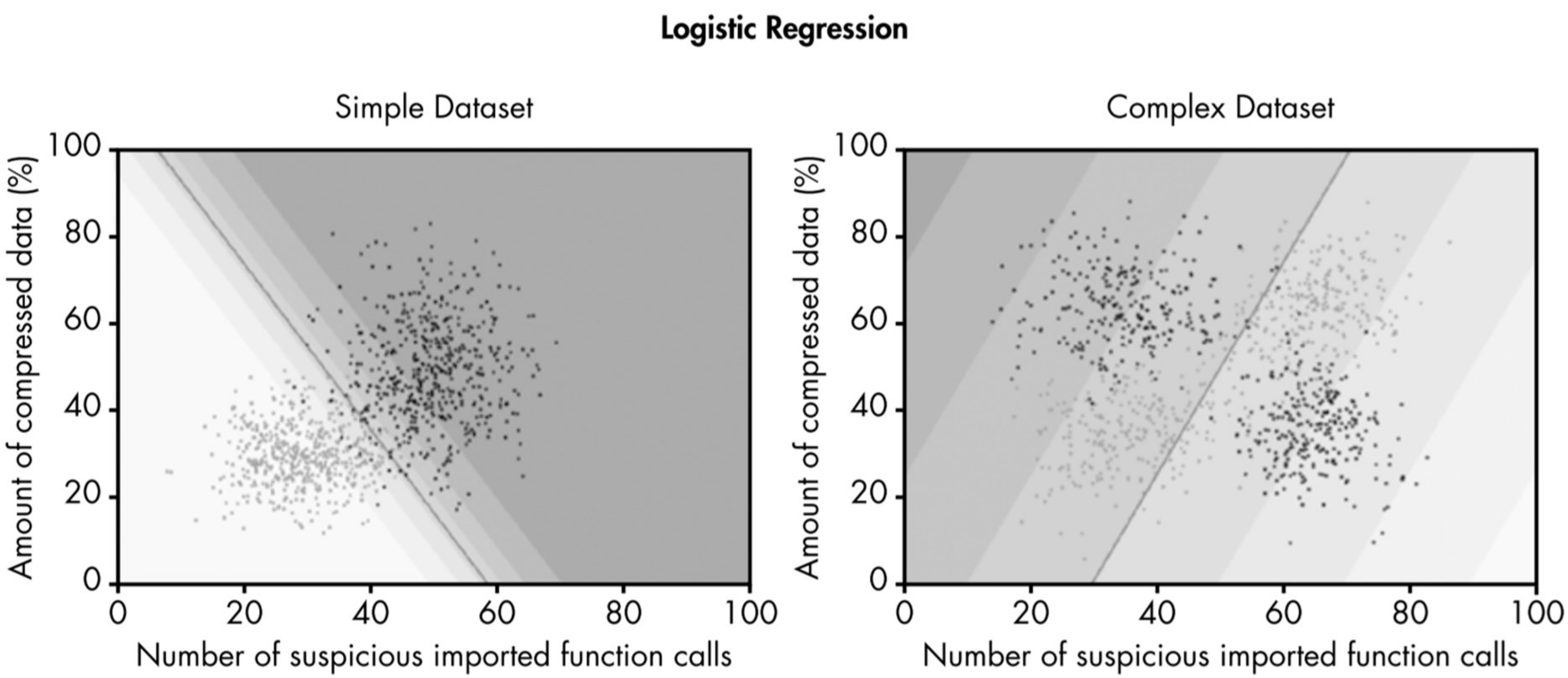


what Makes Models Good or Bad: overfitting and underfitting



Major types of Machine learning algorithms

- Logistic Regression
  - Linear machine learning algorithm



Major types of Machine learning algorithms

- Logistic Regression

```
def logistic_regression(compressed_data, suspicious_calls, learned_parameters): ❶
compressed_data = compressed_data * learned_parameters["compressed_data_weight"] ❷
    suspicious_calls = suspicious_calls * learned_parameters["suspicious_calls_weight"]
score = compressed_data + suspicious_calls + bias ❸
    return logistic_function(score)

def logistic_function(score): ❹
    return 1/(1.0+math.e**(-score))
```

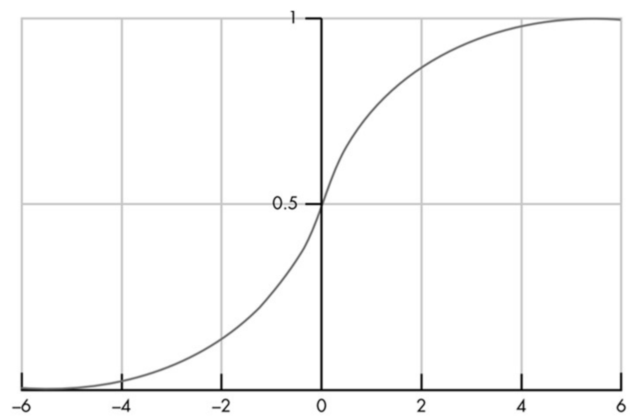


Figure 6-12: A plot of the logistic function used in logistic regression

Major types of Machine learning algorithms

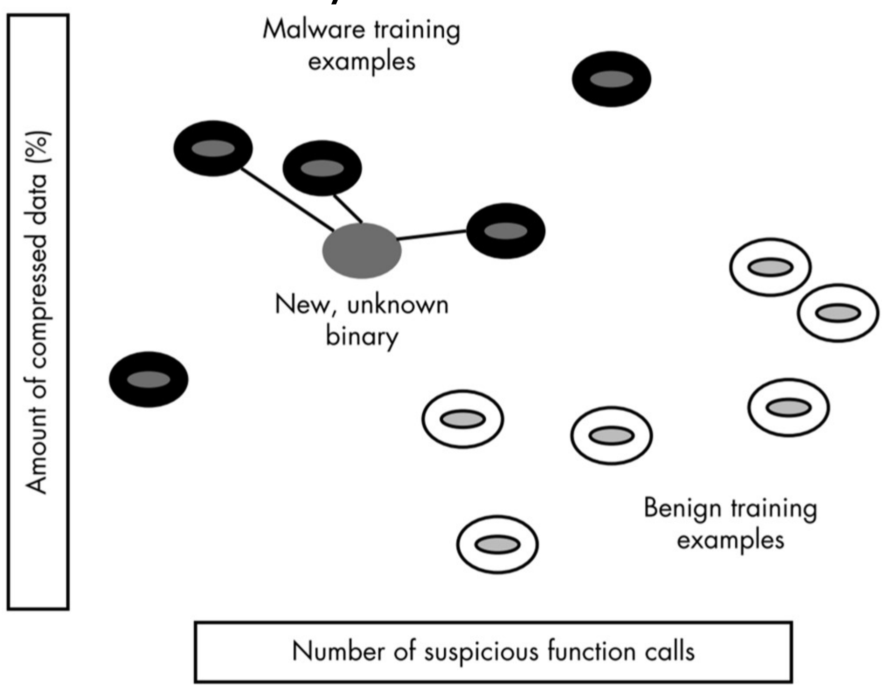
- Logistic Regression
  - how exactly does learn based on the training data?
    - It uses an iterative, calculus-based approach called gradient descent.
    - We won’t get into the details

Major types of Machine learning algorithms

- When to Use Logistic Regression
  - has distinct advantages and disadvantages
    - advantage is that one can easily interpret it
      - Features that have high weight are those the model interprets as malicious
      - Features with negative weight are those the model believes are benignware
      - is a fairly simple approach
    - Disadvantage
      - when the data is complex, logistic regression often fails.

Major types of Machine learning algorithms

- K-Nearest Neighbors
  - the file is malicious
    - if the majority of the k closest binaries to an unknown binary are malicious



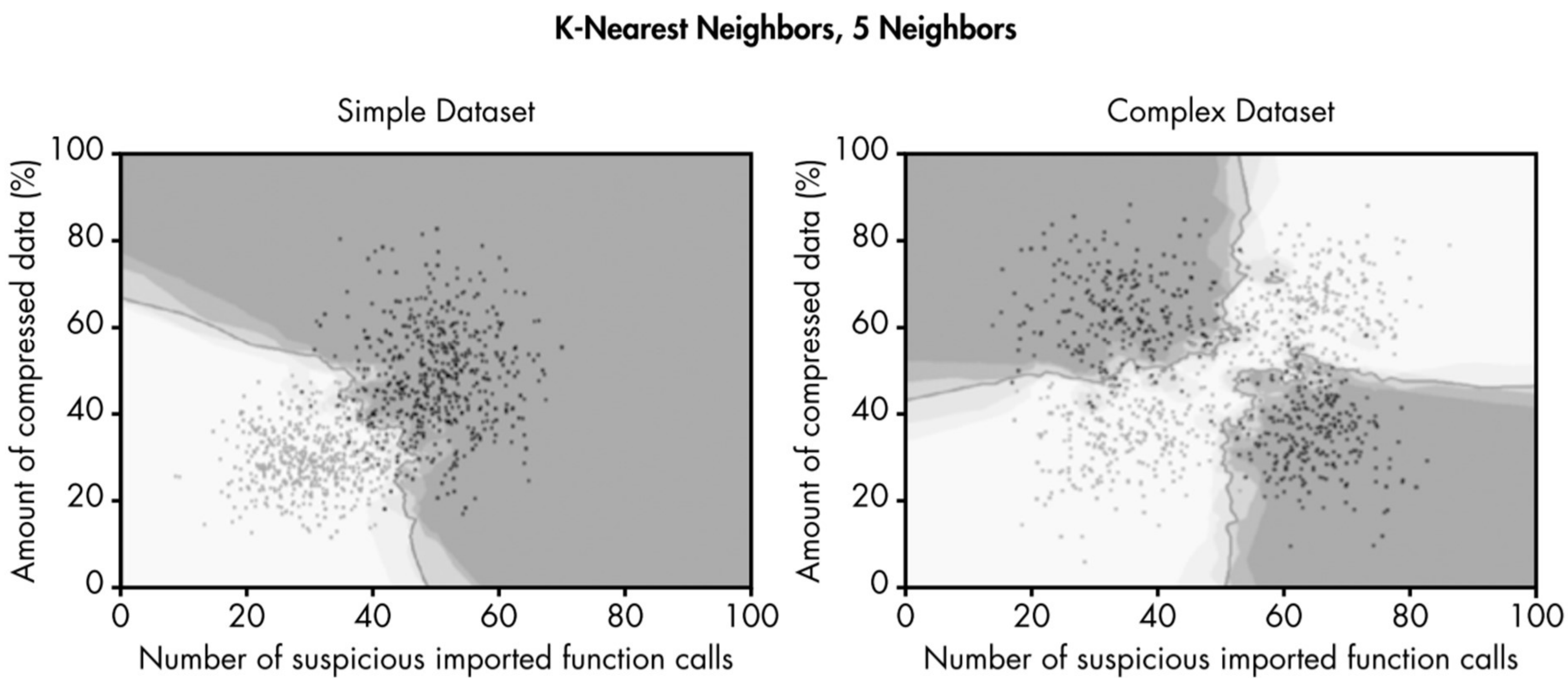
Major types of Machine learning algorithms

- K-Nearest Neighbors
  - most common distance function

```
import math
def euclidean_distance(compression1,suspicious_calls1, compression2, suspicious_calls2): ❶
    comp_distance = (compression1-compression2)**2 ❷
    call_distance = (suspicious_calls1-suspicious_calls2)**2 ❸
    return math.sqrt(comp_distance + call_distance) ❹
```

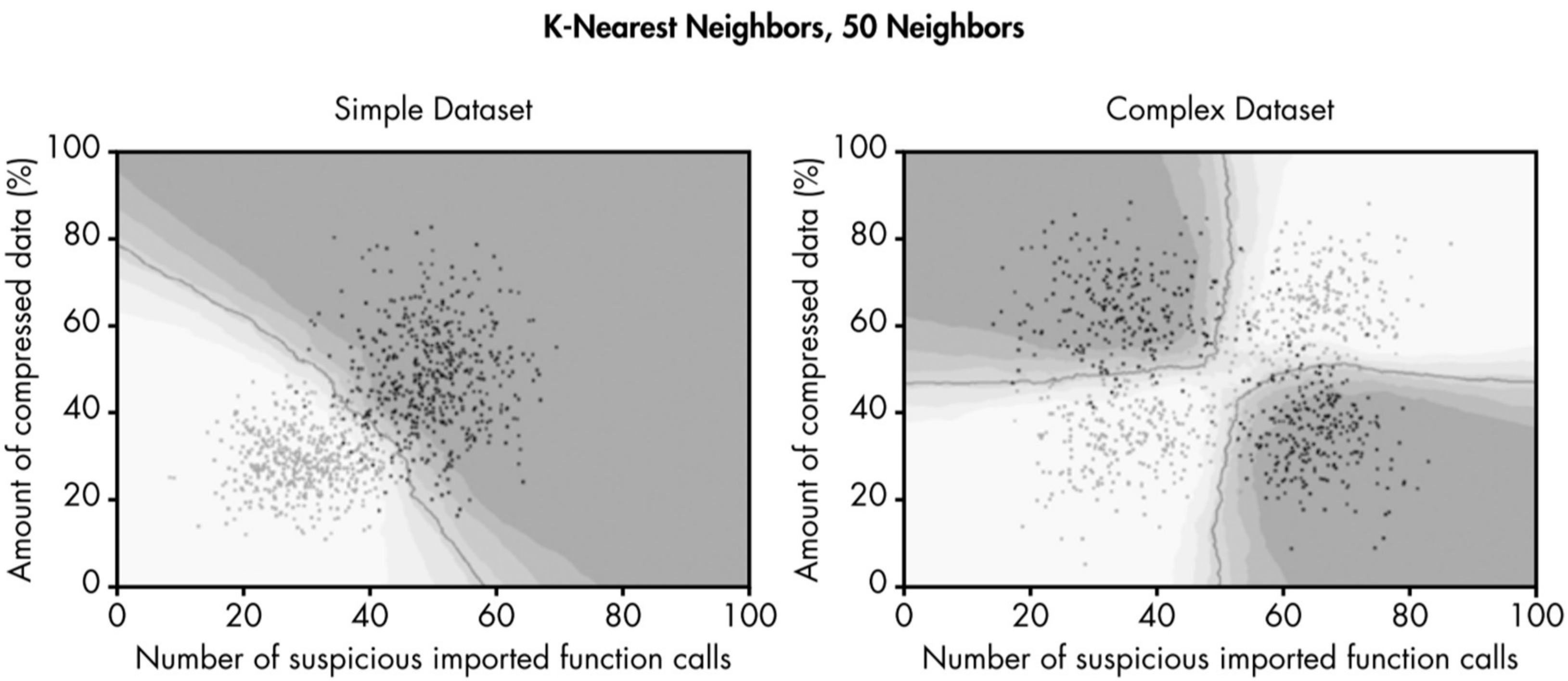
Major types of Machine learning algorithms

- K-Nearest Neighbors
  - Choosing the Number of Neighbors That Vote



Major types of Machine learning algorithms

- K-Nearest Neighbors
  - Choosing the Number of Neighbors That Vote

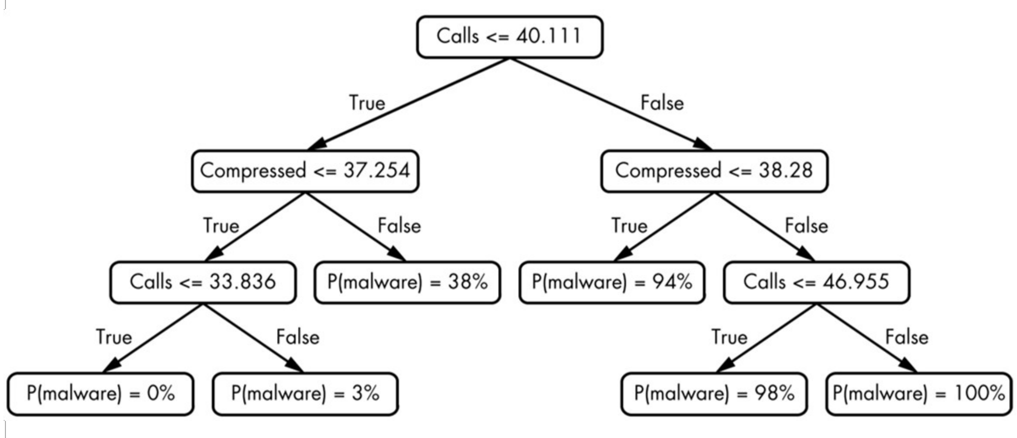


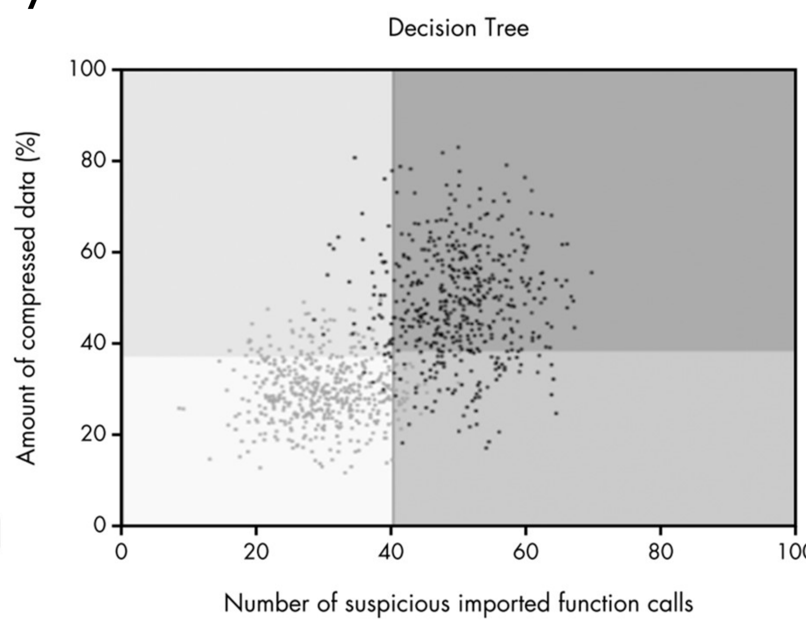
Major types of Machine learning algorithms

- When to Use K-Nearest Neighbors
  - good algorithm when
    - you have data where features don’t map cleanly onto the concept of suspiciousness
    - but closeness to malicious samples is a strong indicator of maliciousness.
  - E.g. when classifying malware families that share code
    - KNN might be a good algorithm to try
  - Another advantage
    - it provides clear explanations of why it has made a given classification decision

Major types of Machine learning algorithms

- Decision Trees
  - automatically generate a series of questions to decide whether or not a given binary is malware
  - similar to the game Twenty Questions





Major types of Machine learning algorithms

- Decision Trees
  - Choosing a Good Root Node
    - best root node is the one for which we get
      - a “yes” answer for most if not all samples of one type
      - a “no” answer for most if not all samples of the other type
  - Picking Follow-Up Questions
    - similar to the root node

Major types of Machine learning algorithms

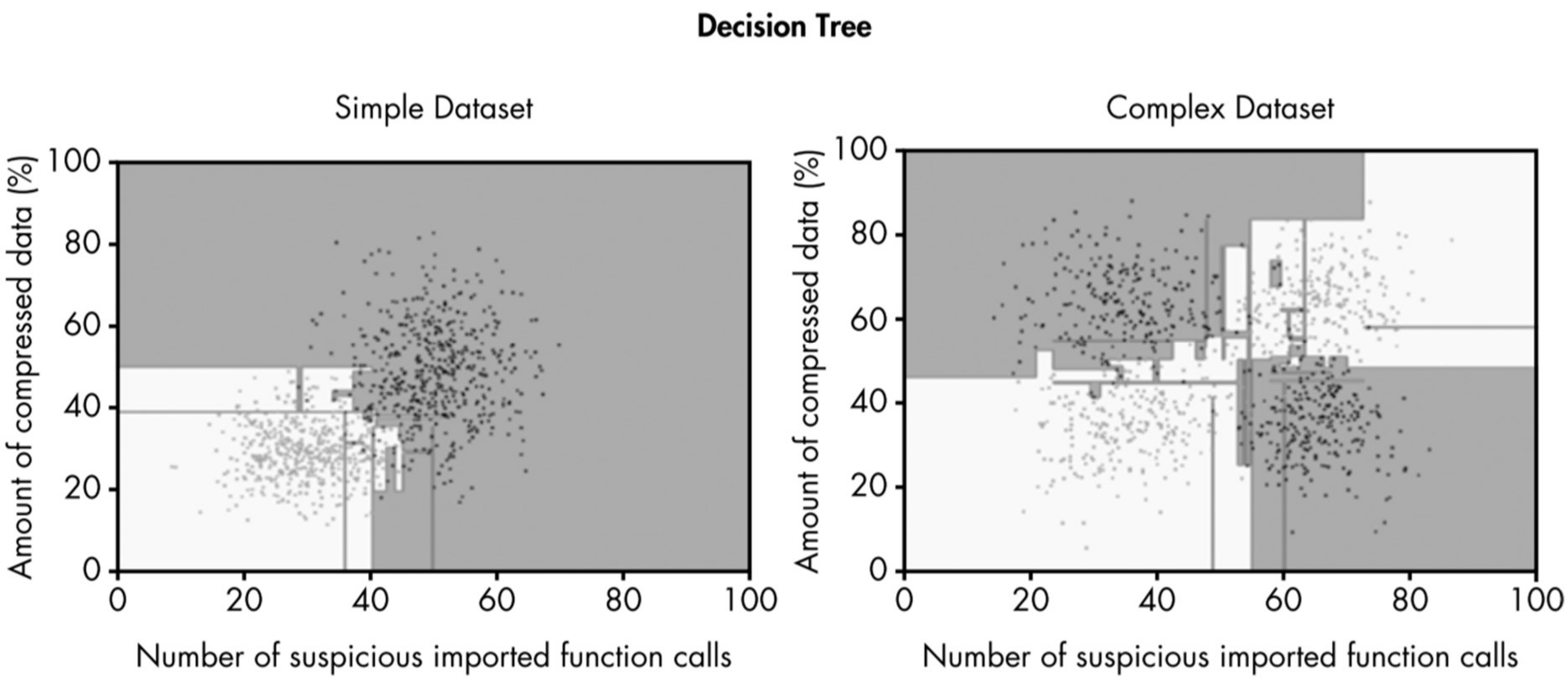
- Decision Trees
  - When to Stop Asking Questions
    - limit the number of questions
    - limit tree depth
    - allow tree to keep growing until it is certain about every example in training set

Major types of Machine learning algorithms

- Decision Trees
  - When to Stop Asking Questions
    - constraining the size of the tree prevents overfitting
    - allowing the tree to grow increases the complexity of the decision boundary
    - Practitioners usually try multiple depths

Major types of Machine learning algorithms

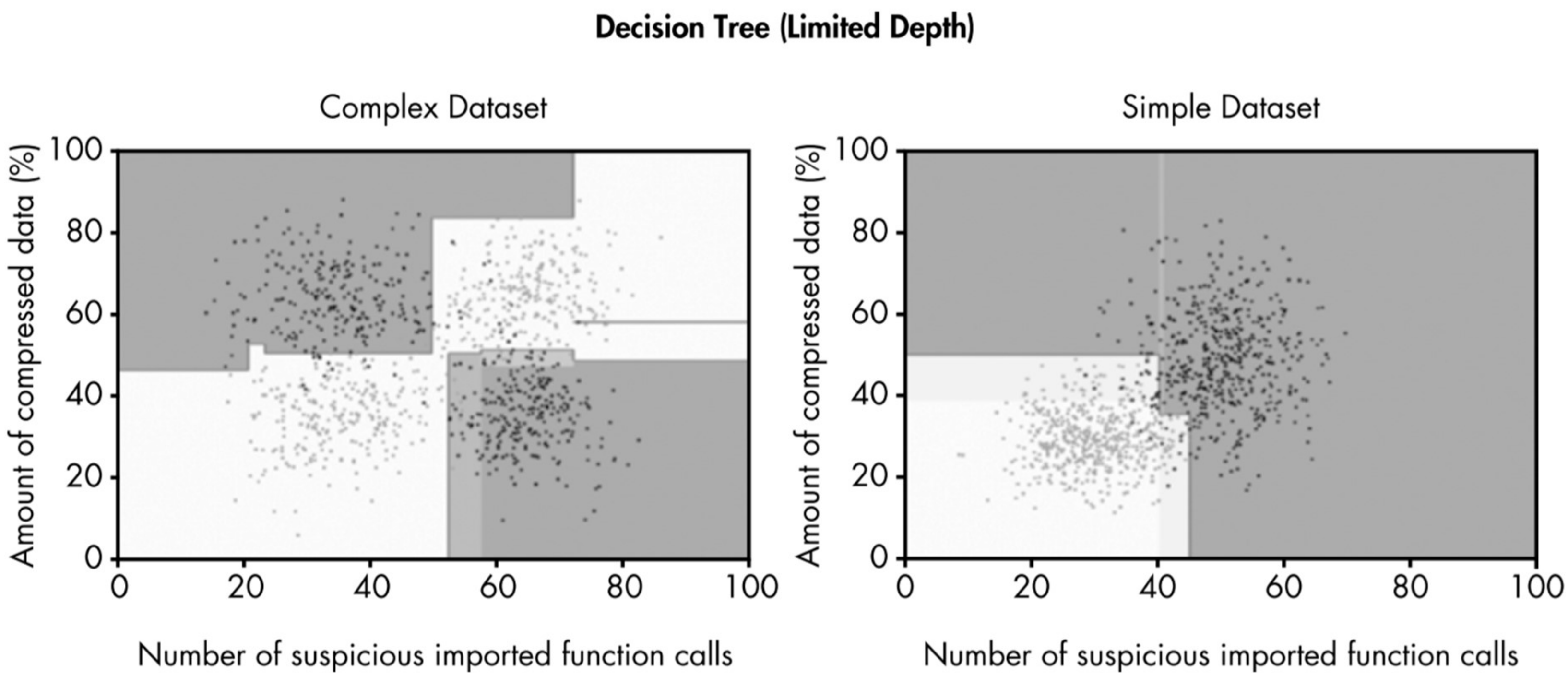
- Decision Trees



No maximum depth

Major types of Machine learning algorithms

- Decision Trees



Depth limited to 5

Major types of Machine learning algorithms

- When to Use Decision Trees
  - they often do not result in very accurate models.
    - The reason is related to their jagged decision boundaries
  - don’t usually learn accurate probabilities around their decision boundaries

Major types of Machine learning algorithms

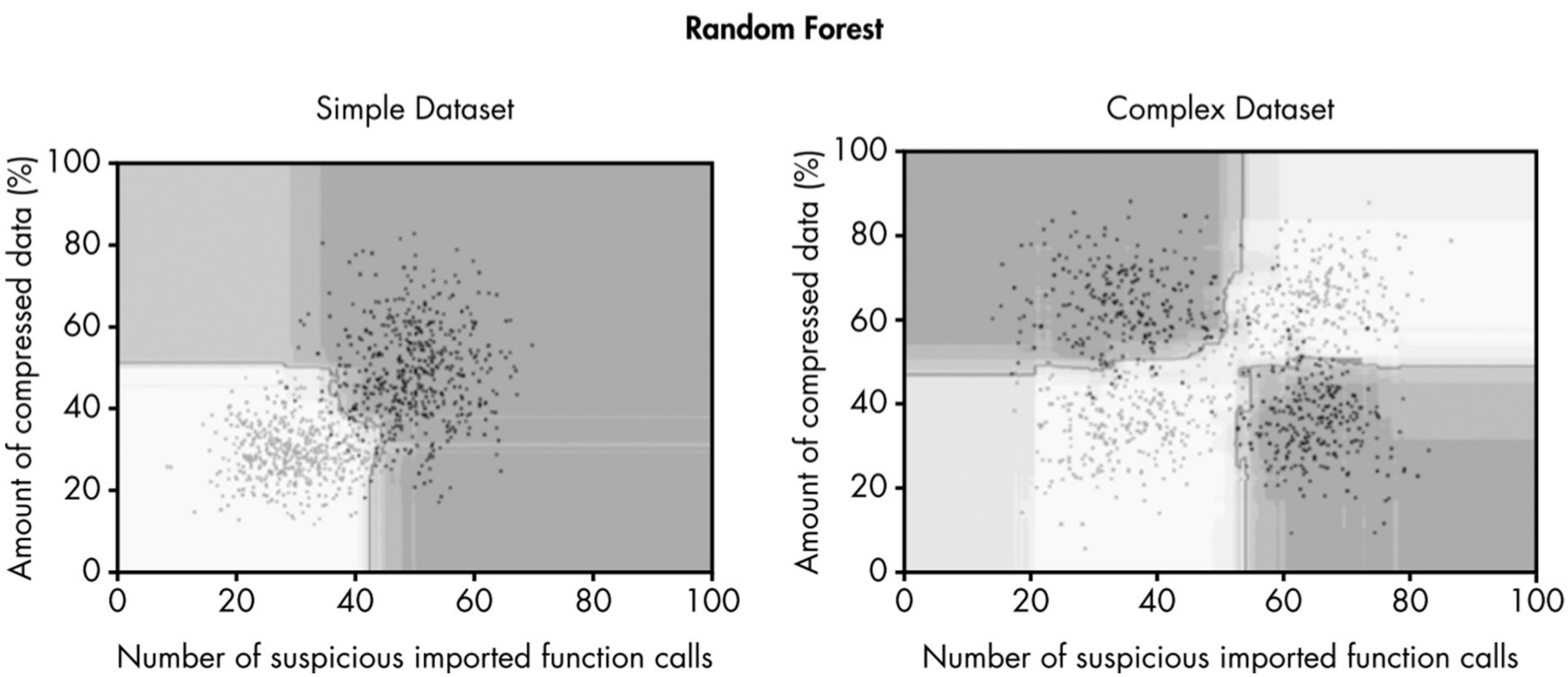
- Random Forest
  - Use hundreds or thousands of decision trees in concert
    - decision trees to vote to decide for a new binary
    - decision trees should be diverse
      - have different perspectives

Major types of Machine learning algorithms

- Random Forest
  - Training for each tree:
    - Randomly choose some examples from training set.
    - Build a decision tree from the random sample.
      - each time ask a question of only a handful of features
      - and disregard the other features.
  - Detection on a previously unseen binary
    - Run detection for each individual tree on the binary.
    - Decide based on the number of trees that voted "yes."

Major types of Machine learning algorithms

- Random Forest



Using 100 decision trees