Excerpted from “A Technological Review of NSA’s SKYNET”, a briefing for U.S. policymakers on the intelligence community’s use of machine-learning algorithms in the identification of potential Al-Qaeda couriers.

SKYNET’s AI

The NSA’s project aimed at locating AQSL couriers is an archetypical application of machine learning for classification. As the agency concluded from preliminary analysis done on confirmed couriers, their targets have a different behavioral profile than do average Pakistanis: They travel more and in different patterns, swap SIM cards more often, and spend more time on the phone than everyone else. A human analyst would be unable to manually discern these subtle trends in a pool of 55 million individuals. A well-trained AI, on the other hand, is perfectly suited to the task.

SKYNET data scientists test drove a number of different machine-learning techniques in their search for an effective model. Each approach, however, followed the methodological framework common to all supervised machine learners: Of the seven known AQSL couriers, six would be mixed in with a random sample of 100,000 presumably ordinary Pakistanis. The program would be told, “These six individuals are known couriers. Find others like them.” Presented with these ground truths, the model would formulate a series of rules to identify the behavioral profile marking the six couriers as unique. The seventh courier, known to the modellers but unknown to the AI, would be mixed in with a different random sample of 100,000 ordinary citizens. The algorithm formulated during the training phase would be tested on this new, unlabelled dataset.

As are those of all machine-learning algorithms, SKYNET’s classification scheme is purely probabilistic. If an individual travelled frequently and in certain patterns, swapped their SIM card often, and made an abnormally large number of phone calls, the AI would assign them a higher probability of being an AQSL courier. An individual who spent every weekend at the beach in Karachi, used the same phone for years in a row, and made only long, infrequent calls to the same few numbers would be assigned a low probability of being an AQSL courier.

Through this iterative process of training and testing models, SKYNET analysts eventually settled on a what is called a “random forest” algorithm. In the last few years, random forests have become a go-to choice for machine-learning applications not only across industry and academia, but also amongst intelligence practitioners.[1] To understand the dynamics of the random forest classifier, however, we have to first grasp the mechanics of its primary component part: the decision tree classifier.

The Decision Tree

A decision tree is a classification algorithm in which the model takes a dataset and divides it into increasingly smaller chunks to try and make each chunk as homogenous as possible.

To illustrate the inner mechanics of a single tree, we’ll use one to classify a small sample from the Iris Database, a popular dataset for testing machine learning algorithms. Of the 100 irises in our small sample, half are of the type *iris virginica* and half are of the type *iris versicolor*. Each flower is described by four features: Petal length, petal width, sepal length, and sepal width. When presented with the training dataset, the tree will formulate a human-readable set of division criteria to determine the type of a randomly chosen specimen.

---

[1] HIMR Data Mining Problem Book

Decision Tree Visualized

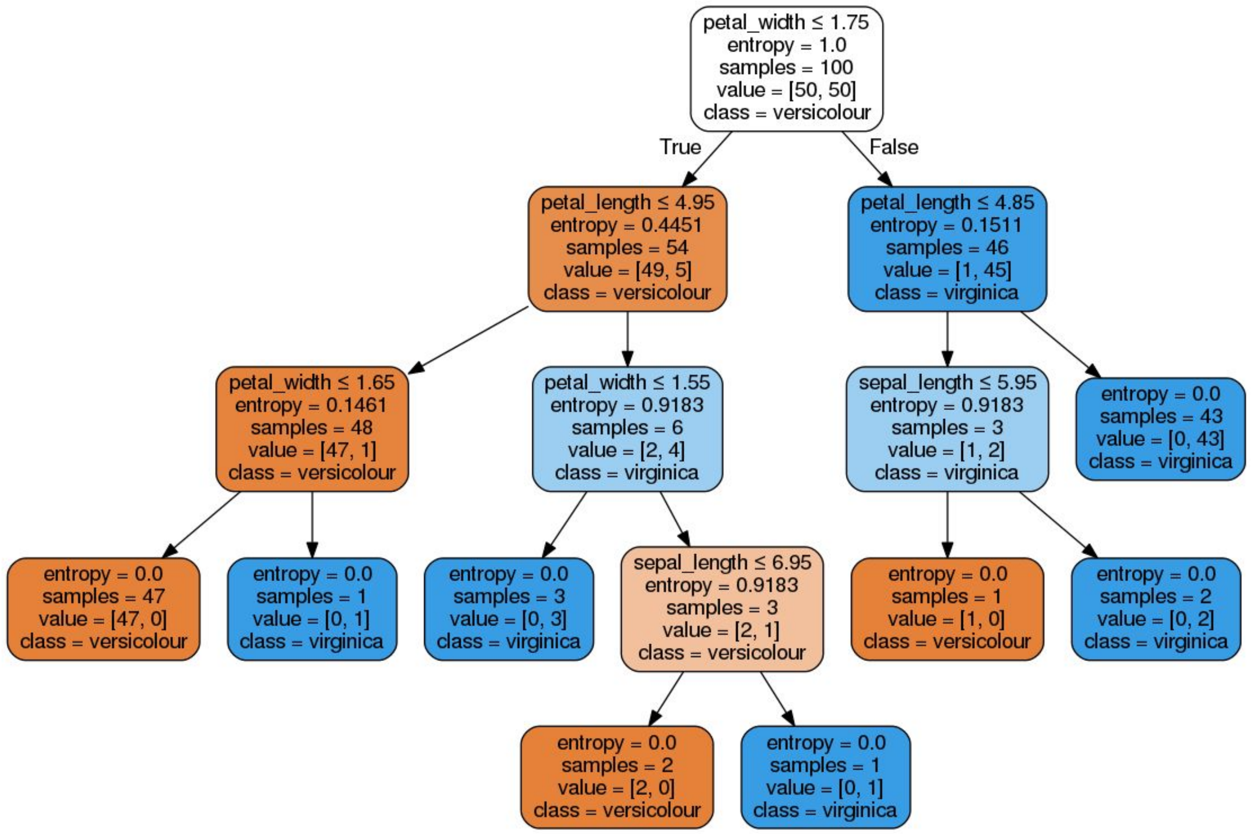

Image by Rudy Gilman

As we can see, the tree first splits the irises into two groups based on petal width, putting all flowers with petal length less than 1.75 cm into the bin on the left and all flowers with petal length greater than 1.75 cm into the bin on the right. It then divides the group on the right by petal length, then again by sepal length. It splits groups into smaller and smaller divisions until the whole dataset had been classified. When given a new specimen, the decision tree can make an educated guess as to its type by following the classification scheme above.

To classify the dataset as quickly and accurately as possible, the tree branches along the feature that maximizes information gain. Of the four features considered, dividing by petal width results in the cleanest split between *versicolor* and *virginica* specimens.

Decision Tree Feature Selection

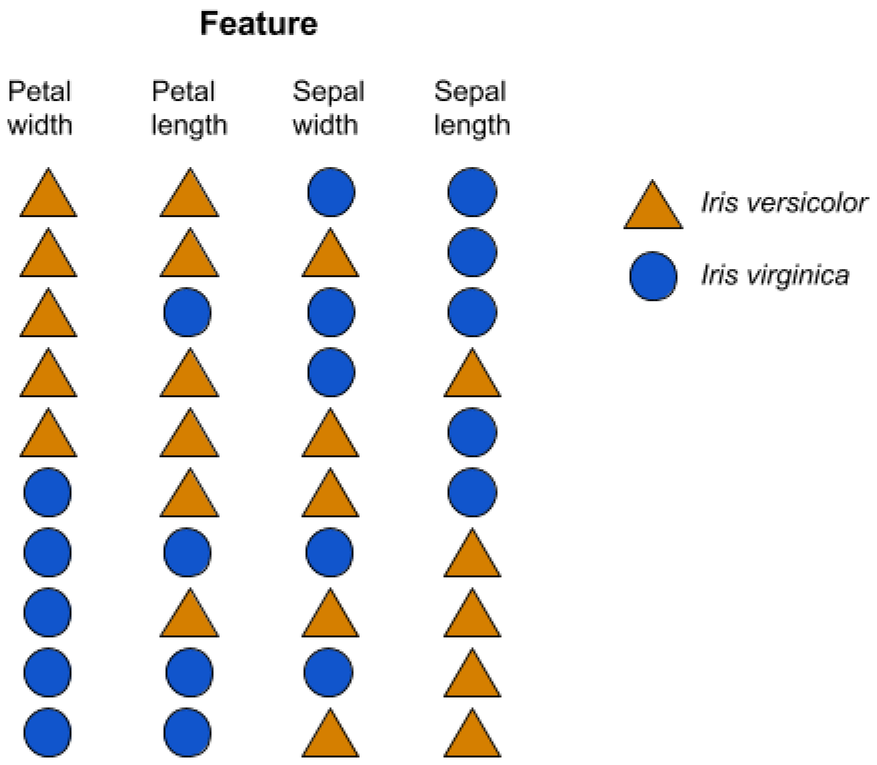

Image by Rudy Gilman

Bias vs. Variability

Although single decision trees have the benefit of being easily interpretable—each algorithm can be diagramed as in the example above—they are prone to overfitting. When allowed to grow to full size, a decision tree will continue to branch until each leaf is perfectly homogenous. An overly-fitted decision tree can perfectly predict the training data; it will do substantially worse than a pruned tree, however, when presented with a specimen it hasn't seen before.

Striking a balance between over- and under-fitting—between bias and variability—is a constant struggle for modellers. Overfitting a model on training data will lead to low bias but high-variability, with the model centered on the correct value but overly sensitive to randomness in the data. Underfitting a model prevents it from capturing the underlying relationships in the data, leading to low variability but high bias.

To illustrate this tradeoff, let’s look at the decision regions produced by four different models working with the Iris Database. To illustrate our point, we’ll throw in another type of iris (*setosa*) and only consider two features: petal length and petal width.

Bias vs. Variability in Model Selection

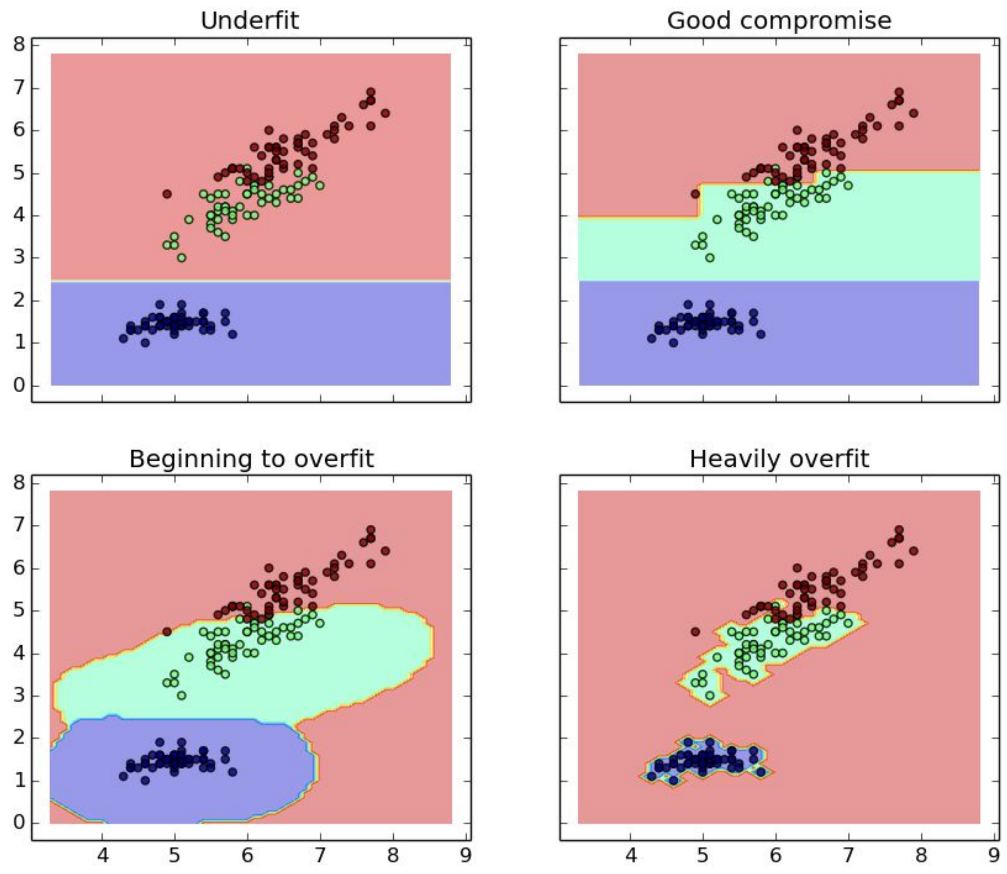

Image by Rudy Gilman

As we can see, the underfit model misses an entire class of iris. The heavily overfit model perfectly predicts this specific training set, but will easily misclassify any new irises that appear. Though the bias-variance compromise model makes a few errors on the training data, it will perform substantially better “in the wild” than the overfit models.

The Random Forest

To increase the stability of results, decrease overfitting and improve predictive power in the face of skewed or missing data—all while maintaining computational efficiency—multiple pruned trees can be combined into an ensemble model. Although each tree in the forest is a ‘“weak learner” whose predictions are only slightly better than random chance, an ensemble of weak learners can be a powerful tool for predictive analytics.

In the most common implementation of a random forest, a number of bootstrapped samples[2] of the original dataset are drawn, with each used to grow a single decision tree. To avoid growing a forest of identical trees, each bootstrapped sample is given only a fraction of the available features (the square root of the total available features is a common starting point). Final classification is conducted by majority vote among the individual trees, sometimes weighted, sometimes not.[3]

A random forest such as that utilized by SKYNET would be composed of hundreds of trees, each given a random 20-30 features. While it took a standard laptop less than a second to compose the simple decision tree shown above, it would take an industrial-strength computer hours to build a forest of decision trees from a sample of 100,000 individuals described by 80 features, such as the those used by the NSA in the construction of SKYNET. Innovative modifications such as “boosting”, “bagging”, and the weighting technique mentioned above are often used to enhance model performance.

[2] In this case, “bootstrap sampling” means sampling with replacement until a sample dataset of the same length as the original has been constructed.
[3] The GCHQ typically weights their trees. It’s reasonable to assume SKYNET modellers do, as well.

Measuring and Optimizing Performance

A binary classifier such as SKYNET or our first iris model can make only two predictions: Courier or not courier, *virginica* or *versicolor*. The prediction can be either true or false. The set of four possible combinations of these outcomes is often visualized as a “confusion matrix”. The tradeoffs facing analysts and modellers can be conceptualized in these terms.

Tradeoffs in Model Selection

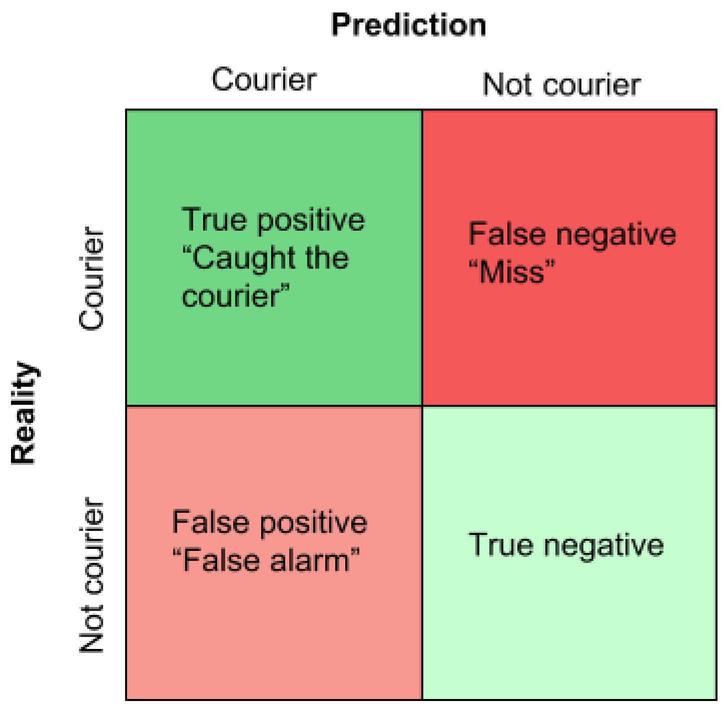

Image by Rudy Gilman

As discussed earlier, models are tested using a process of cross validation, with a random portion of the data withheld for testing purposes and the remainder used to train the program. The process is repeated numerous times, with the average score usually a decent proxy for how the model will perform “in the wild”—assuming the new data is drawn from the same population as the training data.

When using a heavily skewed dataset in which very few targets are mixed in with thousands of non-targets (such as the dataset used by SKYNET), however, using accuracy as a scoring criterion can yield a false sense of confidence—simply predicting that everyone is a nontarget yields an accuracy of over 99%. In these cases, it's helpful to use a metric such as the Receiver Operator Characteristic (ROC)[4], a curve showing the true positive rate at every false positive rate. ROCs are primarily used for binary classification, so to illustrate the technique we'll narrow down our Iris Database again to just *virginica* and *versicolor* and frame the ROC in terms of *virginica* predictions (in a binary model, not predicting *virginica* is the same as predicting *versicolor*).

Measuring Accuracy

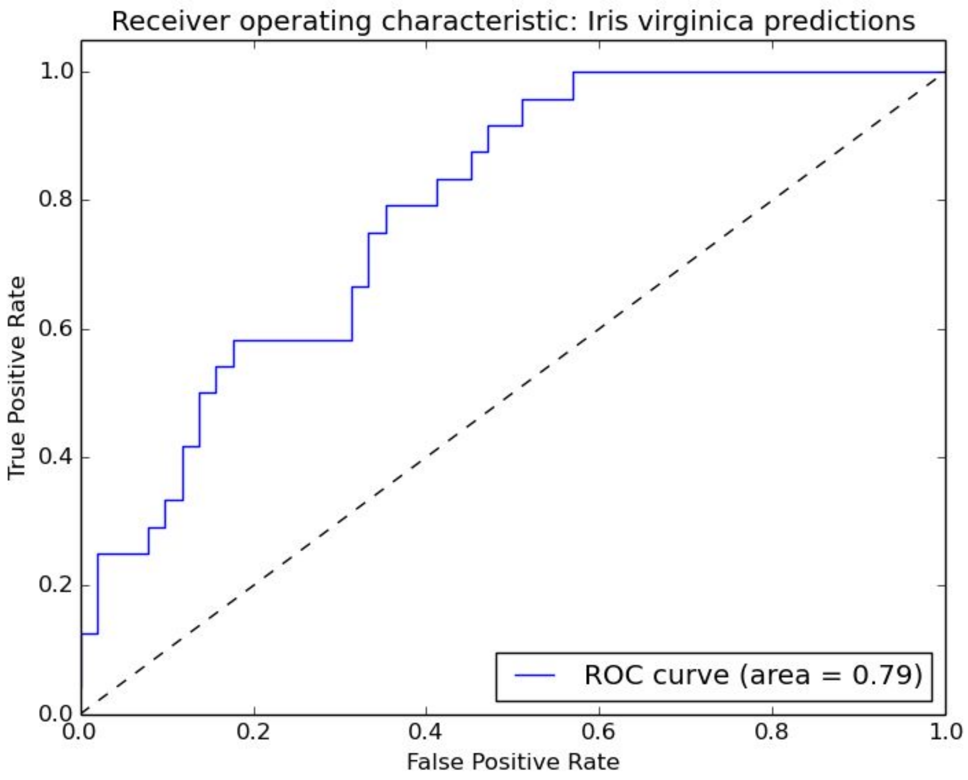

Image by Rudy Gilman

[4] The strange name is a result of its first use as a metric to gauge radar analyses during WWII.

As you can see, our model performs significantly better than random guessing, represented by the dotted diagonal. The ROC of a perfect model would form a 90-degree angle in the top left corner of the graph.

Performance vs. Computing Cost

Optimizing the performance of a model means increasing the area under the curve (AUC) of the ROC by pushing the line up and to the left, increasing the true positive rate at every false alarm rate. Increasing the AUC is a technical exercise, with the modeller balancing model performance against computational expense. As discussed earlier with regards to SKYNET, modellers will typically test-drive multiple models to find the one that maximizes performance for a given task at a given computational budget. Once a model is selected, hyperparameters (such as the number of features to include in a tree, for example) can be further tweaked to maximize the AUC.

False Alarms vs. Misses

Once the model has been optimized from a technical standpoint, the next tradeoff is faced by the analyst. With no more technical gains to be squeezed from a model, every increase in predictive power will be offset by a corresponding increase in false alarms. Conversely, setting a lower threshold to flag fewer false positives means failing to flag real targets. No matter how good the model, users will always be forced to choose between 1) casting a wide net and sifting through a pile of false alarms to avoid missing targets, and 2) casting a smaller net and accepting misses as the cost of avoiding a deluge of false alarms. In stats-speak, this is the classic tradeoff between type-I and type-II errors.

In business, accepting a high false-alarm rate in exchange for a low miss rate means sending diaper coupons to a non-pregnant household, recommending a movie the viewer might not actually like, or diverting normal email correspondence to the spam folder. In intelligence applications, however, the stakes are significantly higher. As discussed earlier, no spy agency able to correctly spell the word “courier” would mistake every individual given a high score by their probabilistic classifier as an actual, confirmed AQSL courier; the manpower resources required to investigate each lead generated by the model, however, are still quite costly. As the HIMR Problem Book notes, “tolerance for false positives is very low: if an analyst is presented with three leads to look at, one of which is probably of interest, then they might have the time to follow that up. If they get a list of three hundred, five of which are probably of interest, then that is not much use to them."