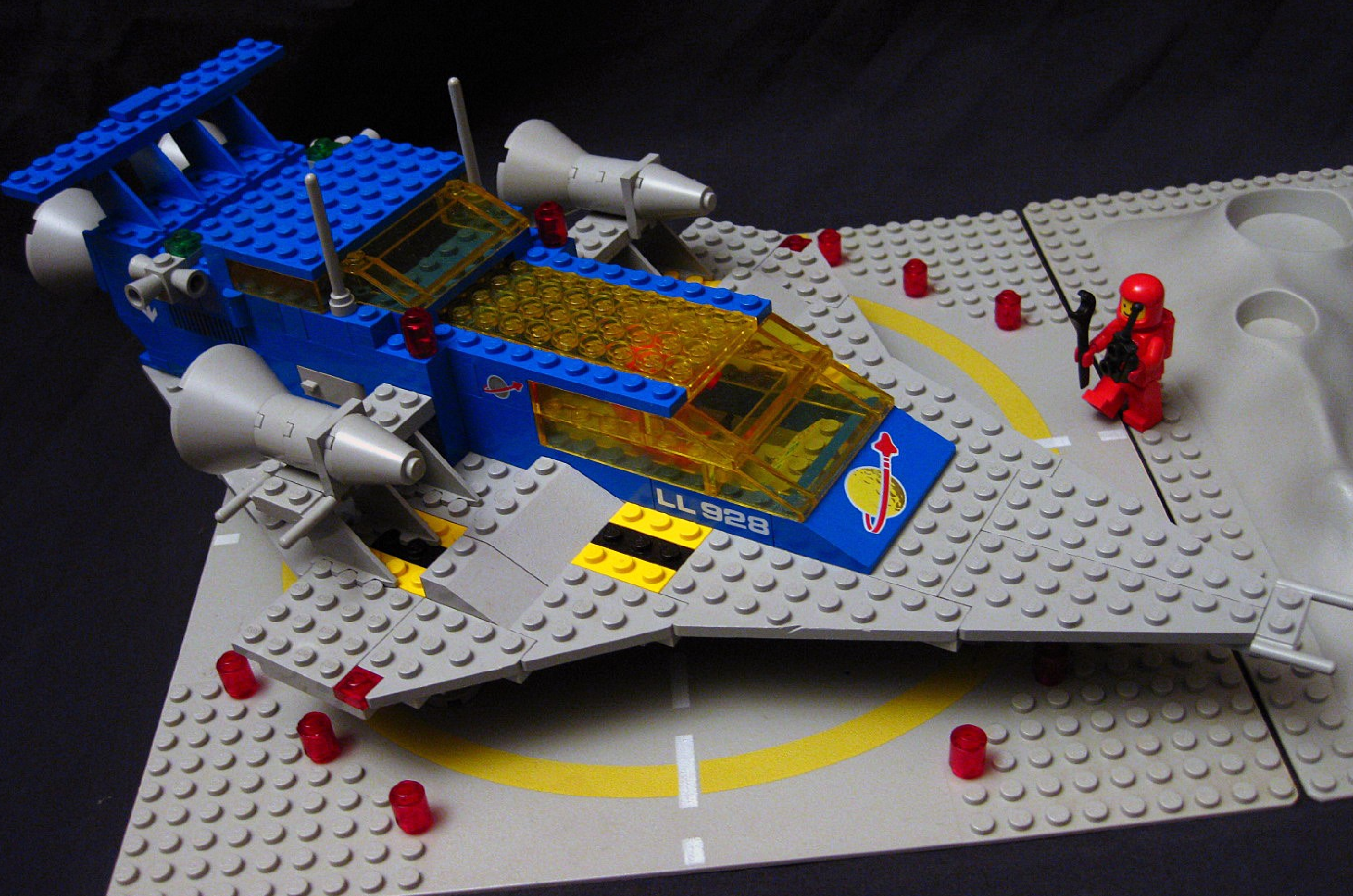
COMBINING MODELS



Material

- Mehta et al notebooks:
  - 08 Bagging
  - 09 Random Forests
  - 10 XGBoost
- Our XGBoost notebooks

Combining models

- “wisdom of the crowds”

  (if no correlated deficiencies)

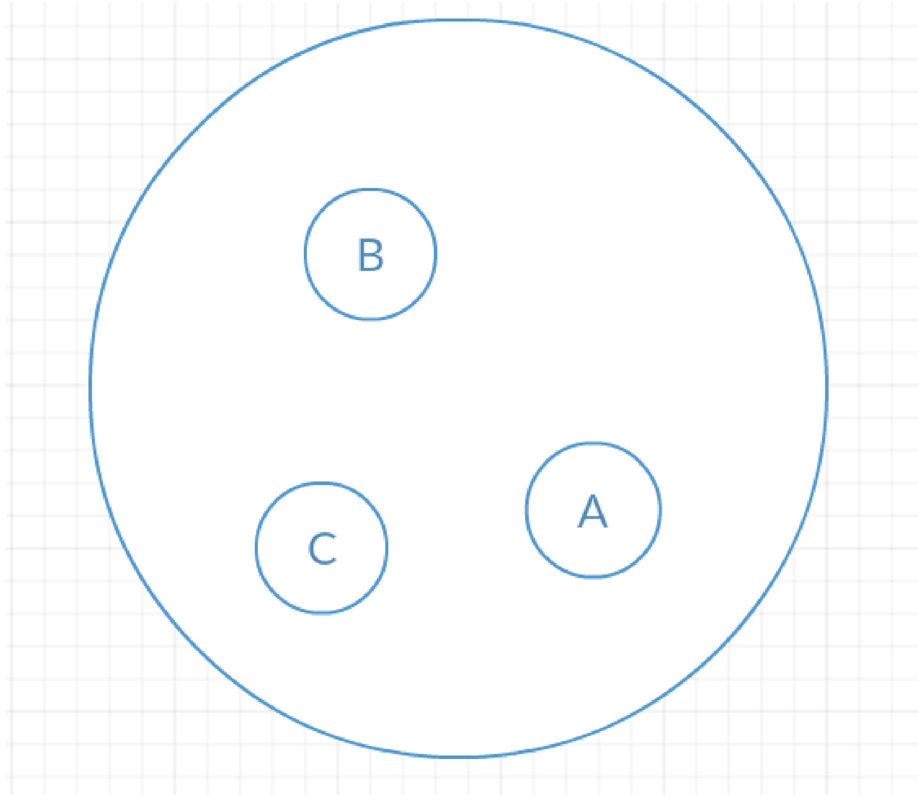
> One of the most powerful and widely-applied ideas in modern machine learning is the use of ensemble methods that combine predictions from multiple, often weak, statistical models to improve predictive performance (Diet-

- Many weak models need less assumptions than a single complicated model
- Today: widely applied techniques

Boosting Algorithms: AdaBoost, Gradient Boosting and XGBoost

Rohith Gandhi Follow
May 6, 2018 · 5 min read

https://hackernoon.com/boosting-algorithms-adaboost-gradient-boosting-and-xgboost-f74991cad38c

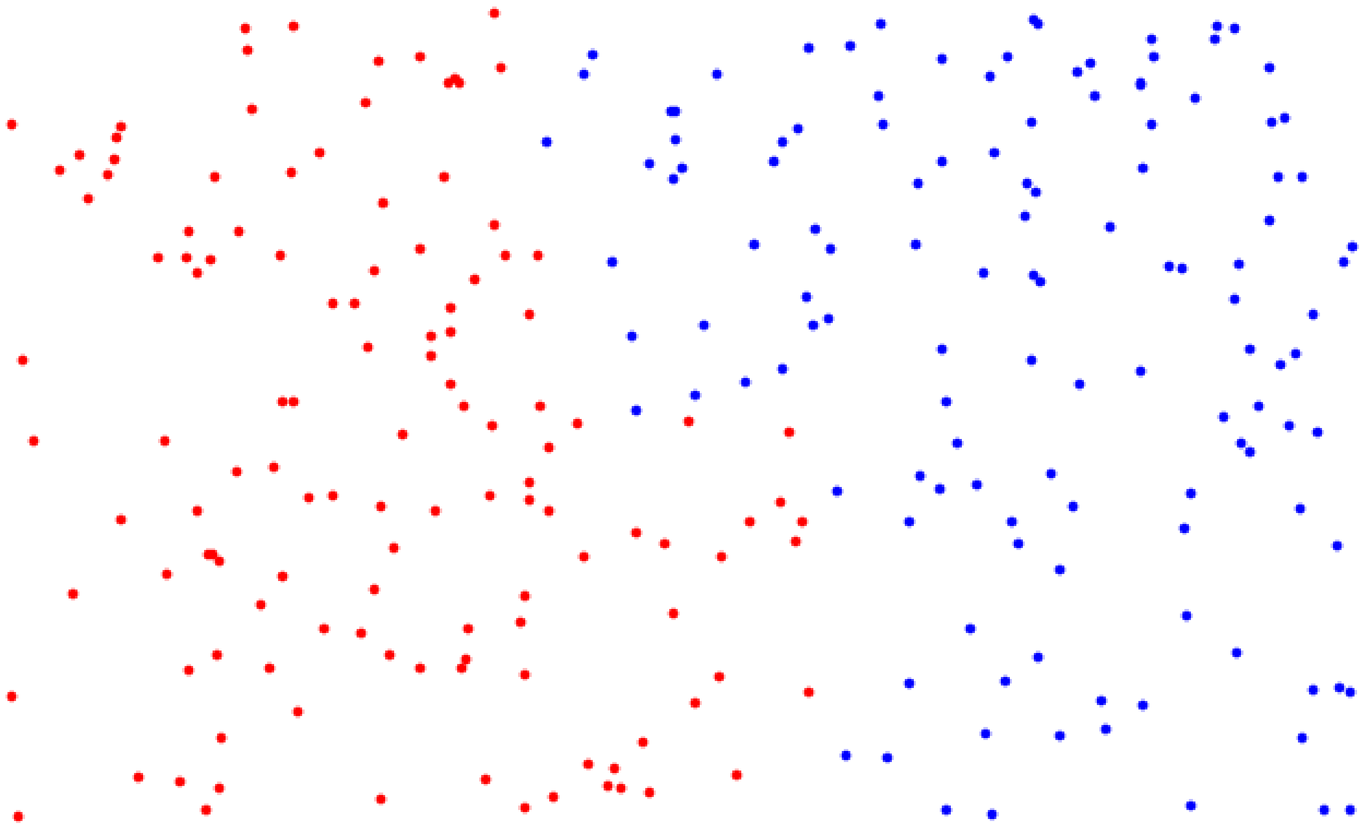


Here, let A, B and C be different classifiers. Their area A represents where the classifier A misclassifies(goes wrong) and area B represents where the classifier B misclassifies and area C represents where the classifier C misclassifies. Since there is no correlation between the errors of each classifier, combining them and using a technique of democratic voting to classify each object, this family of classifiers will never go wrong.

Correlated vs uncorrelated models

- Important that combined models are uncorrelated (See the review)
- Correlated models → not so better variance, possibly higher bias
- M uncorrelated models → much reduced variance (~1/M), same bias of single model

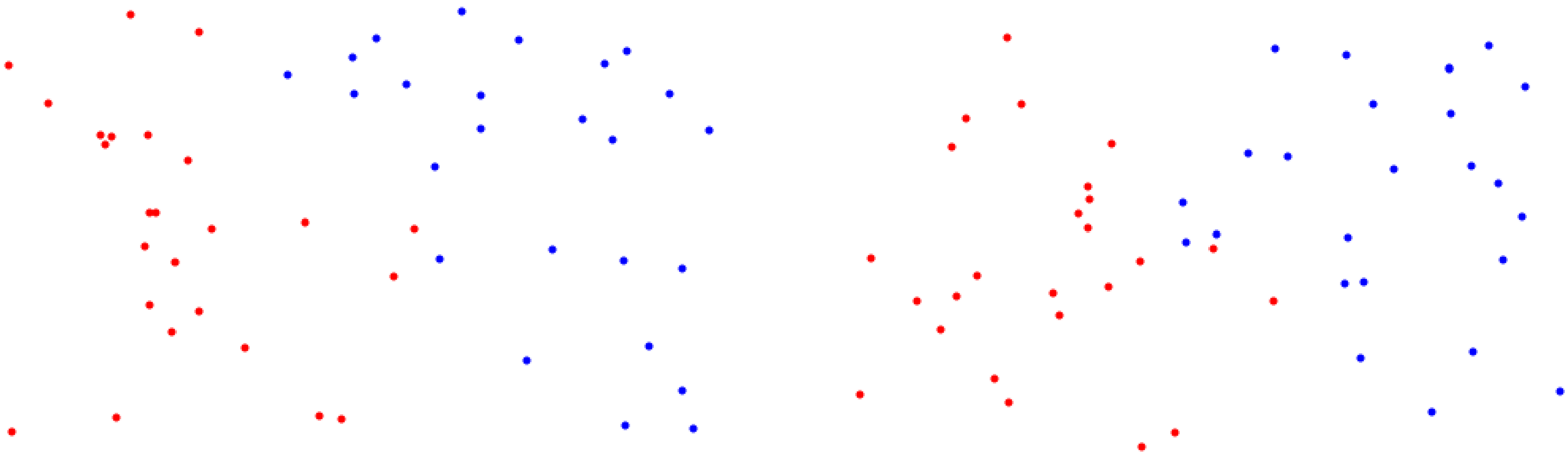
Is there a combination of linear classifiers that predicts the color of these data?



1) bagging

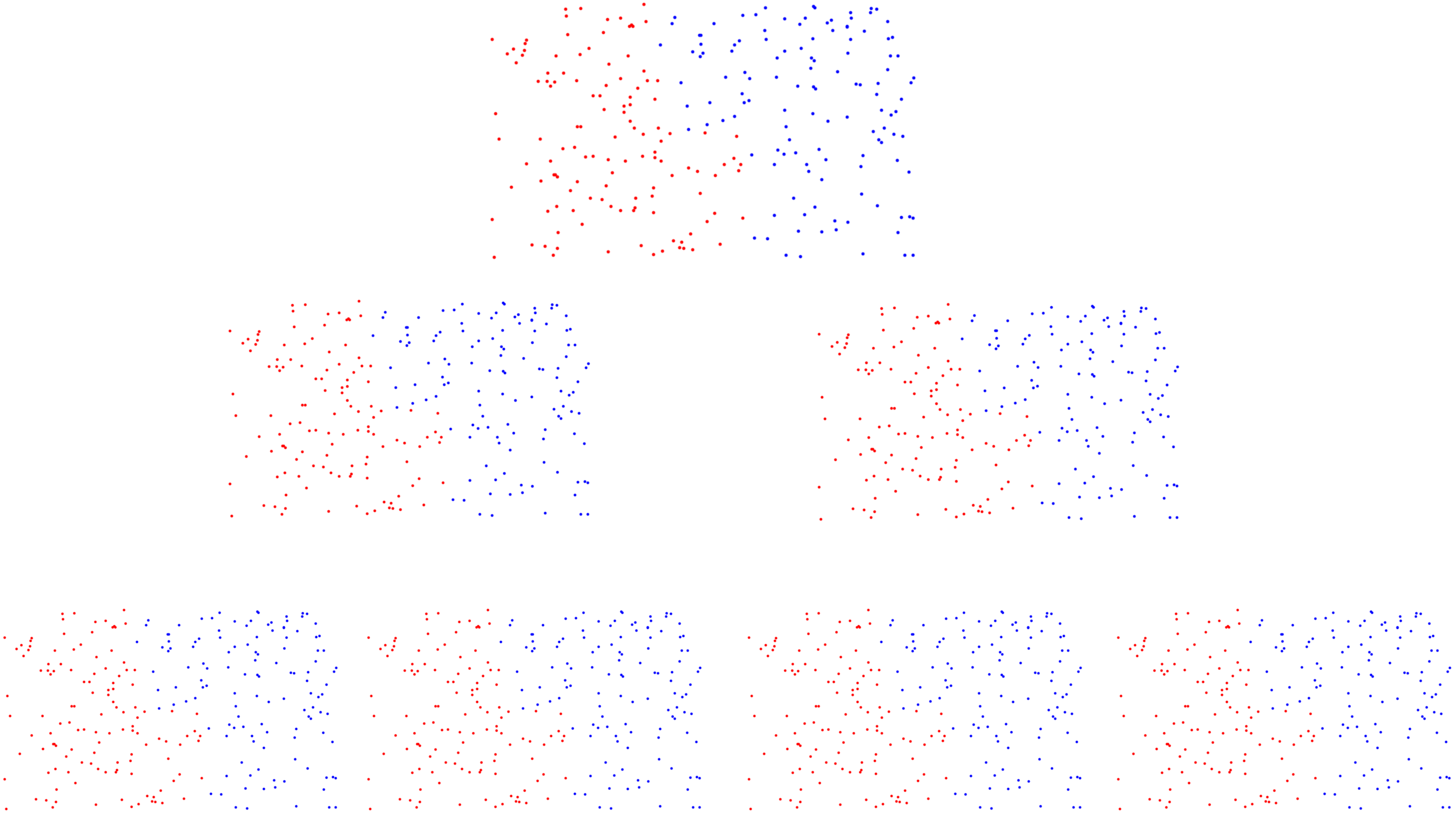
Notebook 8: Bagging a simple binary classifier

- subsample data randomly, many times: bootstrapping



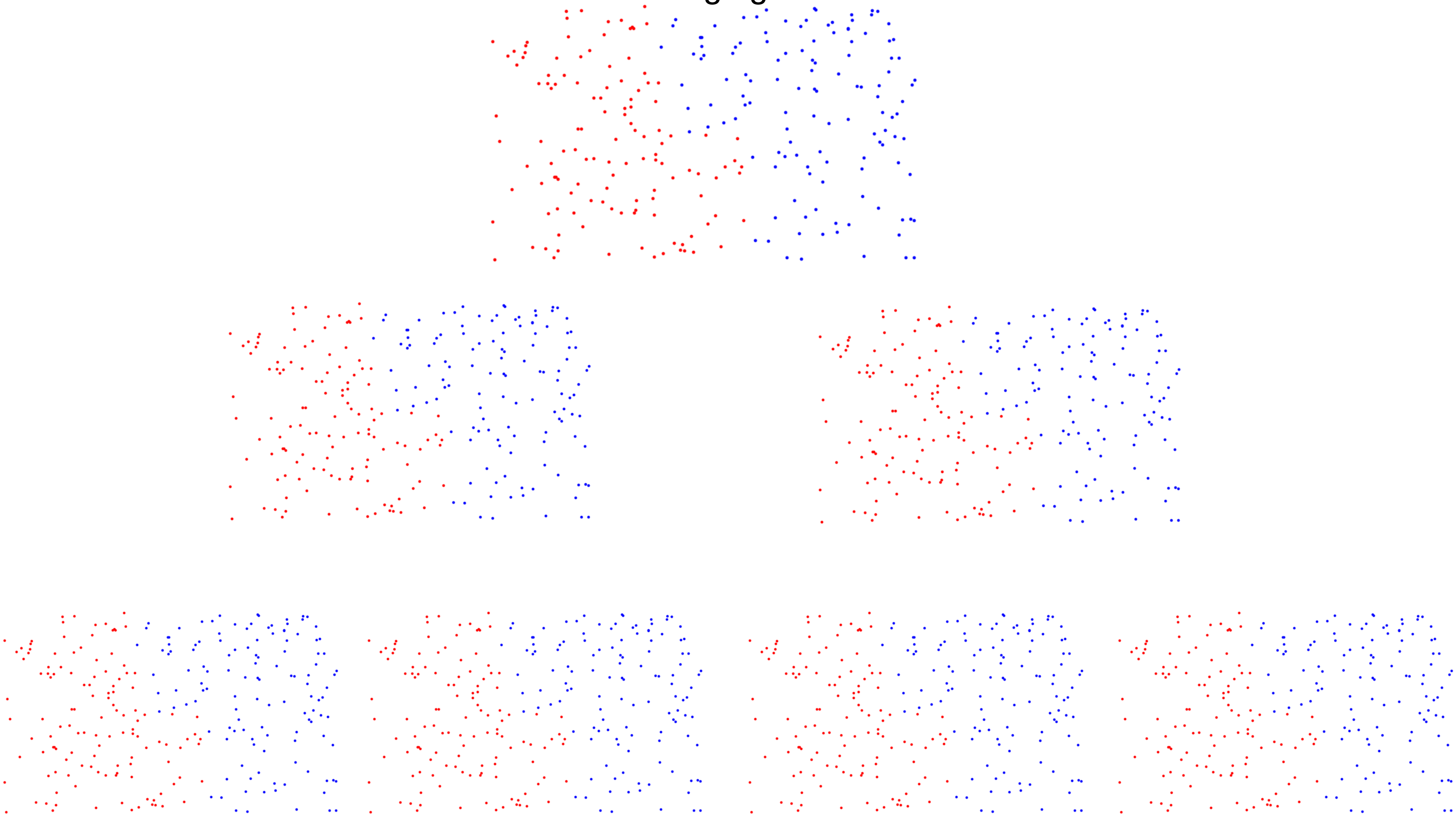
- apply a simple model to each subset
- Combine outputs of models with majority rule
- https://datasciencelab.wordpress.com/2014/01/10/machine-learning-classics-the-perceptron/

2) decision trees

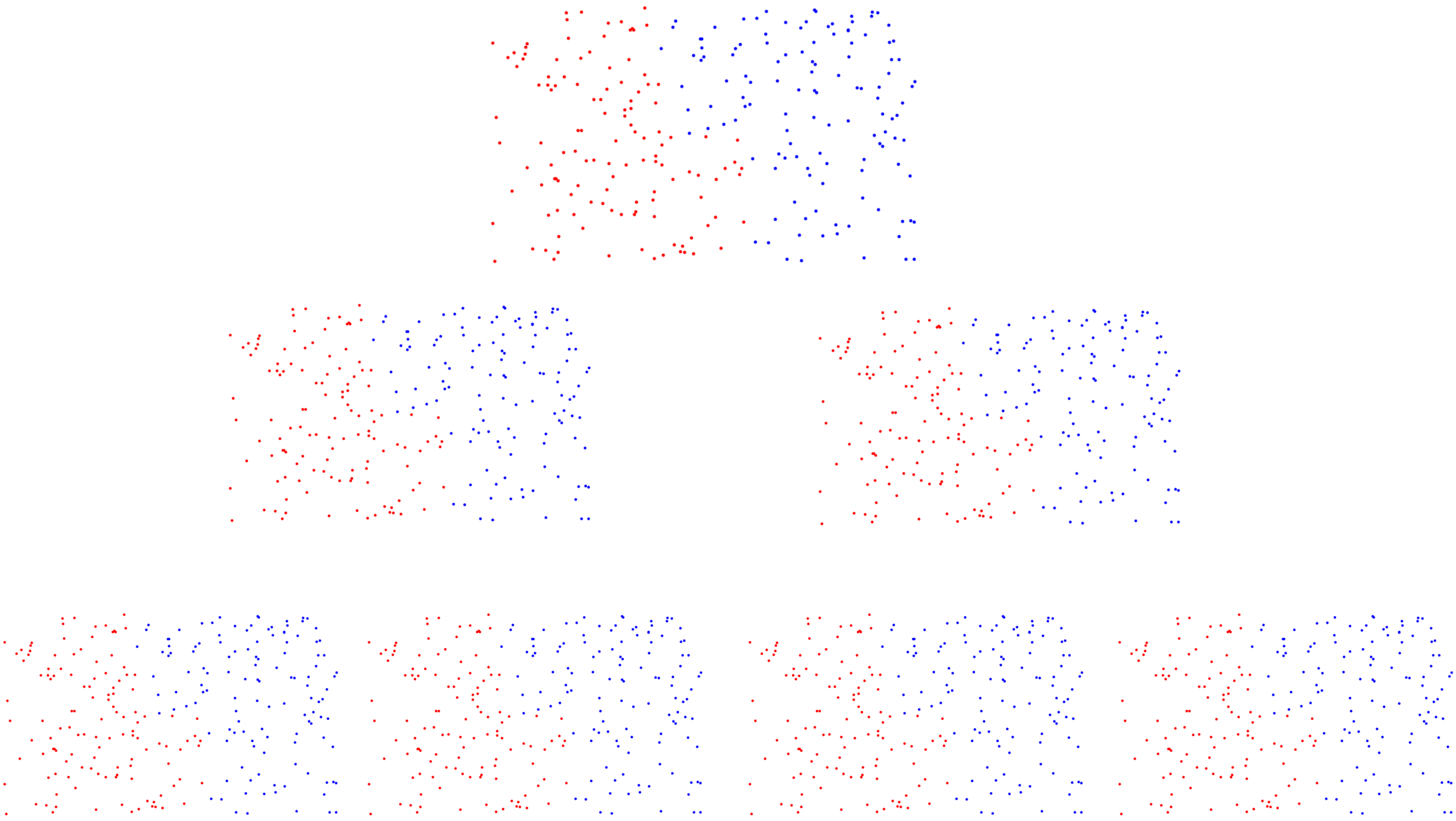


2) decision trees

Changing order



2) decision trees



3) boosting (AdaBoost)

- Form aggregate classifier iteratively
- works on improving the areas where the base learner fails
- https://web.stanford.edu/~hastie/Papers/AdditiveLogisticRegression/alr.pdf

---

Discrete AdaBoost [Freund and Schapire (1996b)]

1. Start with weights $w_i = 1/N$, $i = 1, \ldots, N$.
2. Repeat for $m = 1, 2, \ldots, M$:

   (a) Fit the classifier $f_m(x) \in \{-1, 1\}$ using weights w_i on the training data.

   (b) Compute $\mathrm{err}_m = E_w[1_{(y \neq f_m(x))}]$, $c_m = \log((1 - \mathrm{err}_m)/\mathrm{err}_m)$.

   (c) Set $w_i \leftarrow w_i \exp[c_m 1_{(y_i \neq f_m(x_i))}]$, $i = 1, 2, \ldots, N$, and renormalize so that $\sum_i w_i = 1$.

3. Output the classifier $\mathrm{sign}[\sum_{m=1}^{M} c_m f_m(x)]$.

---

3) boosting (AdaBoost)

Real AdaBoost

1. Start with weights $w_i = 1/N$, $i = 1, 2, \ldots, N$.
2. Repeat for $m = 1, 2, \ldots, M$:

   (a) Fit the classifier to obtain a class probability estimate $p_m(x) = \hat{P}_w(y = 1|x) \in [0, 1]$, using weights w_i on the training data.

   (b) Set $f_m(x) \leftarrow \frac{1}{2} \log p_m(x)/(1 - p_m(x)) \in R$.

   (c) Set $w_i \leftarrow w_i \exp[-y_i f_m(x_i)]$, $i = 1, 2, \ldots, N$, and renormalize so that $\sum_i w_i = 1$.

3. Output the classifier $\text{sign}[\sum_{m=1}^{M} f_m(x)]$.

ALGORITHM 2. *The Real AdaBoost algorithm uses class probability estimates $p_m(x)$ to construct real-valued contributions $f_m(x)$.*

* Drawback: easily defeated by noisy data,
the algorithm tries to fit every point perfectly, hence outliers are a problem