Modelling

What are we going to learn today?

- Ensemble Methods
  - The Three Sources of Error
  - Bootstrap Aggregation (Bagging)
  - Random Forest
  - Boosting
- Cross-Validation in Finance
  - Why K-Fold Cross-Validation Fails in Finance
  - Purged K-Fold Cross-Validation

Ensemble Method

The Three Sources of Errors

Consider a training set of observations $\{x_i\}_{i=1,\ldots,n}$ and real-valued outcomes $\{y_i\}_{i=1,\ldots,n}$. Suppose a function $f[x]$ exists, such that $y = f[x] + \varepsilon$, where ε is white noise with $\mathrm{E}[\varepsilon_i] = 0$ and $\mathrm{E}[\varepsilon_i^2] = \sigma_\varepsilon^2$. We would like to estimate the function $\hat{f}[x]$ that best fits $f[x]$, in the sense of making the variance of the estimation error $\mathrm{E}[(y_i - \hat{f}[x_i])^2]$ minimal (the mean squared error cannot be zero, because of the noise represented by σ_ε^2). This mean-squared error can be decomposed as

$$\mathrm{E}\left[(y_i - \hat{f}[x_i])^2\right] = \left(\underbrace{\mathrm{E}[\hat{f}[x_i] - f[x_i]]}_{bias}\right)^2 + \underbrace{\mathrm{V}[\hat{f}[x_i]]}_{variance} + \underbrace{\sigma_\varepsilon^2}_{noise}$$

An ensemble method is a method that combines a set of weak learners, all based on the same learning algorithm, in order to create a (stronger) learner that performs better than any of the individual ones. Ensemble methods help reduce bias and/or variance.

Bootstrap Aggregation (Bagging) (1/2)

Consider a bagging classifier that makes a prediction on *k* classes **by majority voting** among *N* independent classifiers. We can label the predictions as {0,1}, where 1 means a correct prediction. The accuracy of a classifier is the probability *p* of labeling a prediction as 1. A sufficient condition for considering the classifier "informed" is that the sum of these labels is $X > N/2$. However, given that there are *k* classes, a necessary (non-sufficient) condition for considering the classifier "informed" is that $X > N/k$, which occurs with probability

$$\mathrm{P}\left[X > \frac{N}{k}\right] = 1 - \mathrm{P}\left[X \leq \frac{N}{k}\right] = 1 - \sum_{i=0}^{\lfloor N/k \rfloor} \binom{N}{i} p^i (1-p)^{N-i}$$

The implication is that for a sufficiently large *N*, say $N > p\left(p - \frac{1}{k}\right)^{-2}$, then $p > \frac{1}{k} \Rightarrow \mathrm{P}\left[X > \frac{N}{k}\right] > p$, hence the bagging classifier's accuracy exceeds the average accuracy of the individual classifiers. **This is a strong argument in favor of bagging any classifier in general, when computational requirements permit it**.

Bootstrap Aggregation (Bagging) (2/2)

$$
\begin{aligned}
\mathrm{V}\left[\frac{1}{N}\sum_{i=1}^{N}\varphi_i[c]\right] &= \frac{1}{N^2}\sum_{i=1}^{N}\left(\sum_{j=1}^{N}\sigma_{i,j}\right) = \frac{1}{N^2}\sum_{i=1}^{N}\left(\sigma_i^2 + \sum_{j\neq i}^{N}\sigma_i\sigma_j\rho_{i,j}\right) \\
&= \frac{1}{N^2}\sum_{i=1}^{N}\left(\bar{\sigma}^2 + \underbrace{\sum_{j\neq i}^{N}\bar{\sigma}^2\bar{\rho}}_{\substack{=(N-1)\bar{\sigma}^2\bar{\rho} \\ \text{for a fixed } i}}\right) = \frac{\bar{\sigma}^2 + (N-1)\bar{\sigma}^2\bar{\rho}}{N} \\
&= \bar{\sigma}^2\left(\bar{\rho} + \frac{1-\bar{\rho}}{N}\right)
\end{aligned}
$$

where $\sigma_{i,j}$ is the covariance of predictions by estimators i,j; $\sum_{i=1}^{N}\bar{\sigma}^2 = \sum_{i=1}^{N}\sigma_i^2 \Leftrightarrow \bar{\sigma}^2 = N^{-1}\sum_{i=1}^{N}\sigma_i^2$; and $\sum_{j\neq i}^{N}\bar{\sigma}^2\bar{\rho} = \sum_{j\neq i}^{N}\sigma_i\sigma_j\rho_{i,j} \Leftrightarrow \bar{\rho} = (\bar{\sigma}^2 N(N-1))^{-1}\sum_{j\neq i}^{N}\sigma_i\sigma_j\rho_{i,j}$.

Random Forest

- Decision trees are known to be prone to overfitting, which increases the variance of the forecasts.
- In order to address this concern, the random forest (RF) method was designed to produce ensemble forecasts with lower variance.
- RF shares some similarities with bagging, in the sense of **independently training individual estimators over bootstrapped subsets of the data**.
- The key difference with bagging is that random forests incorporate a second level of randomness: **When optimizing each node split, only a random subsample (without replacement) of the attributes will be evaluated**, with the purpose of further de-correlating the estimators.
- Advantages:
  - Like bagging, RF reduces forecasts' variance without overfitting (remember, as long as $\bar{p} < 1$).
  - RF evaluates feature importance on-the-fly.
  - RF provides out-of-bag accuracy estimates, however in financial applications they are likely to be inflated.
- Caveat:
  - Like bagging, RF will not necessarily exhibit lower bias than individual decision trees.

Boosting

Schapire [1990] demonstrated that we can combine weak estimators (where $p \ll \frac{1}{k}$) in order to achieve one with high accuracy. In general terms, the procedure works as follows:

1. Generate one training set by random sampling with replacement, according to some sample weights (initialized with uniform weights).
2. Fit one estimator using that training set.
3. If the single estimator achieves an accuracy greater than the acceptance threshold on the testing set (e.g., 50% in a binary classifier, so that it performs better than chance), the estimator is kept, otherwise it is discarded.
4. Give more weight to misclassified observations, and less weight to correctly classified observations. Repeat the previous steps until *N* estimators are produced.
5. The ensemble forecast is the *weighted* average of the individual forecasts from the *N* models, where the weights are determined by the accuracy of the individual estimators.

There are many boosting algorithms, of which AdaBoost is one of the most popular (Geron [2017]).

Numerai's tournament

Numerai organizes weekly tournaments where researchers produce 1-month financial forecasts.

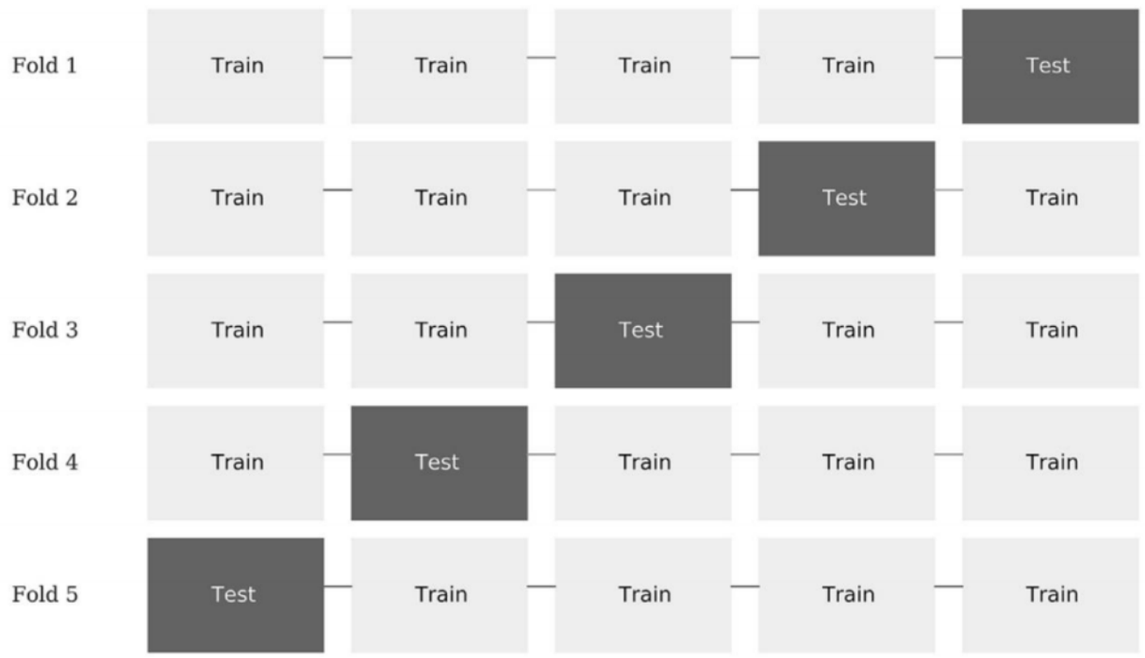
- The dataset is encrypted across company names and features names, so that
  - Vendors allow the distribution of the data
  - Researchers cannot use the testing set for training purposes
- The dataset is structured for cross-sectional studies, to prevent the mapping of companies across time, and defeat the encryption
  - Only relative levels are provided, with monthly frequency, without lags
  - This purposely prevents the modelling of time series characteristics, such as ECM.

When implementing ensemble models on this dataset, it makes sense to:

- **Balance performance across months**: Find month where a model performs poorly, exclude those months from model, and develop a model specifically for those (a kind of boosting).
- **Balance importance across features**: Avoid models that rely heavily on a few features. If those features cease to work, the model will perform poorly.
- **Balance performance across targets**: A robust model will predict different targets.
- **Calibrate bag size**: When bagging, form small bags while controlling that draws come from different months. That reduces the correlation across bags, hence reducing the error's variance.

Cross-Validation in Finance

Why K-Fold Cross-Validation Fails in Finance



Leakage takes place when the training set contains information that also appears in the testing set. Consider a serially correlated feature *X* that is associated with labels *Y* that are formed on overlapping data:

- Because of the serial correlation, $X_t \approx X_{t+1}$.
- Because labels are derived from overlapping datapoints, $Y_t \approx Y_{t+1}$.

Therefore, by placing *t* and *t + 1* in different sets, information is leaked: When a classifier is first trained on (X_t, Y_t), and then it is asked to predict $\mathrm{E}[Y_{t+1}|X_{t+1}]$ based on an observed X_{t+1}, this classifier is more likely to achieve $Y_{t+1} = \mathrm{E}[Y_{t+1}|X_{t+1}]$ even if *X* is an irrelevant feature.

Note that, for leakage to take place, it must occur that $(X_{train}, Y_{train}) \approx (X_{test}, Y_{test})$, and it does not suffice that $X_{train} \approx X_{test}$ or even $Y_{train} \approx Y_{test}$.

Purged K-Fold CV

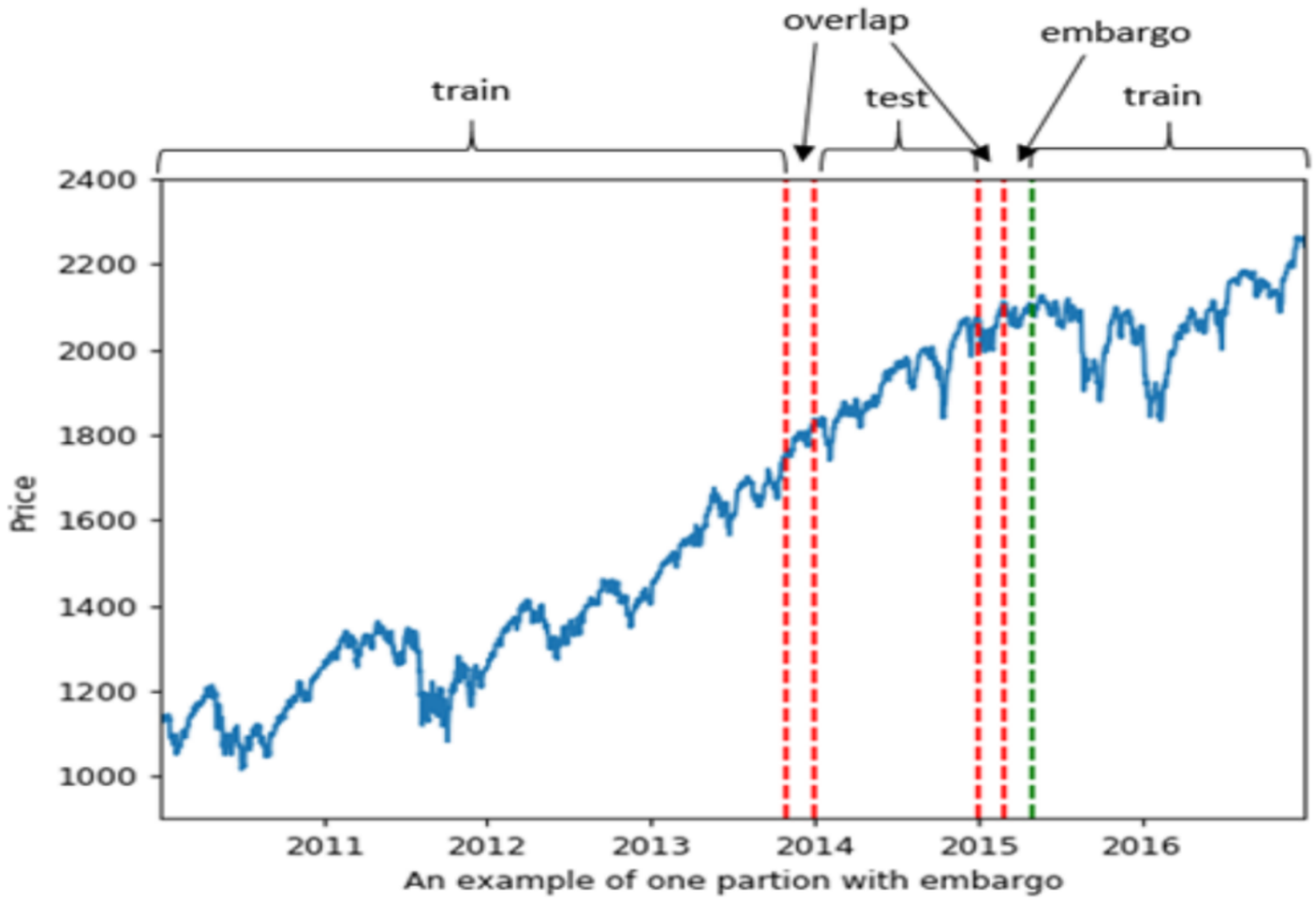
- One way to reduce leakage is to purge from the training set all observations whose labels overlap in time with those labels included in the testing set. I call this process *purging*.
- Consider a label Y_j that is a function of observations in the closed range $t \in [t_{j,0}, t_{j,1}]$, $Y_j = f\left[[t_{j,0}, t_{j,1}]\right]$.
  - For example, in the context of the triple barrier labeling method, it means that the label is the sign of the return spanning between price bars with indices $t_{j,0}$ and $t_{j,i}$, that is $\text{sgn}\left[r_{t_{j,0},t_{j,1}}\right]$.
- A label $Y_i = f\left[[t_{j,0}, t_{j,1}]\right]$ overlaps with Y_j if any of the three sufficient conditions is met:

$$t_{j,0} \leq t_{i,0} \leq t_{j,1}; t_{j,0} \leq t_{i,1} \leq t_{j,1};\ t_{i,0} \leq t_{j,0} \leq t_{j,1} \leq t_{i,1}$$

Embargoed K-Fold CV

- Since financial features often include series that exhibit serial correlation (like ARMA processes), we should eliminate from the training set observations that immediately follow an observation in the testing set. I call this process *embargo*.
  - The embargo does not need to affect training observations prior to a test, because training labels $Y_i = f\left[\left[t_{i,0}, t_{i,1}\right]\right]$, where $t_{i,1} < t_{j,0}$ (training ends before testing begins), contain information that was available at the testing time $t_{j,0}$.
  - We are only concerned with training labels $Y_i = f\left[\left[t_{i,0}, t_{i,1}\right]\right]$ that take place immediately after the test, $t_{j,1} \leq t_{i,0} \leq t_{j,1} + h$.
- We can implement this embargo period *h* by setting $Y_j = f\left[\left[t_{j,0}, t_{j,1} + h\right]\right]$ before purging. A small value $h \approx .01T$, where T is the number of bars, often suffices to prevent all leakage.

Example: Purging and Embargoing



This plot shows one partition of the K-Fold CV. The test set is surrounded by two train sets, generating two overlaps that must be purged to prevent leakage.

To further prevent leakage, the train observations immediately after the testing set are also embargoed.