THE UNIVERSITY OF TEXAS AT AUSTIN
McCOMBS SCHOOL OF BUSINESS

Machine Learning in Finance

David Puelz

April 27, 2016

Prediction

This will be a lecture on prediction in financial markets.

Prediction and the bias-variance tradeoff

Trees

Application to finance data

The Goal

- Predict a target variable Y with input variables X.
- This is often called supervised learning.

The Goal

- Predict a target variable Y with input variables X.
- This is often called supervised learning.

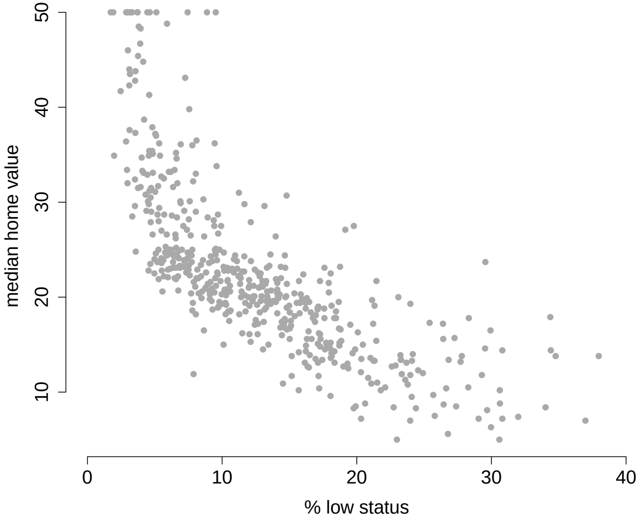
We can frame the problem by supposing Y and X are related in the following way:

$$Y_i = f(X_i) + \epsilon_i$$

To achieve our goal, we need to: *Learn or estimate* $f(\cdot)$ from data.

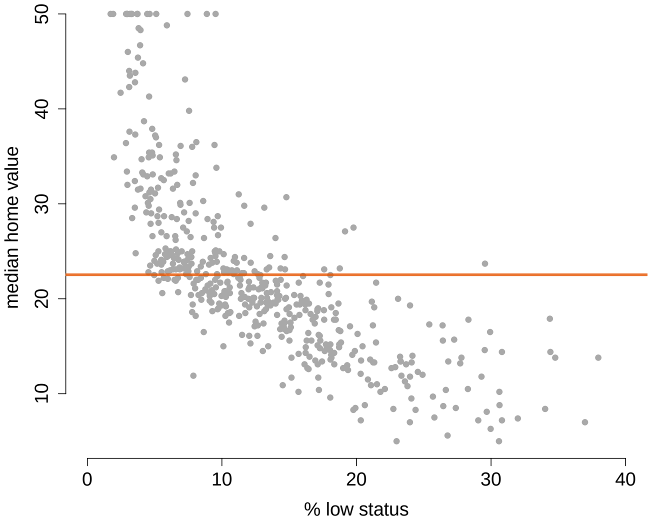
Boston housing data

Predict median home value with percent low economic status.



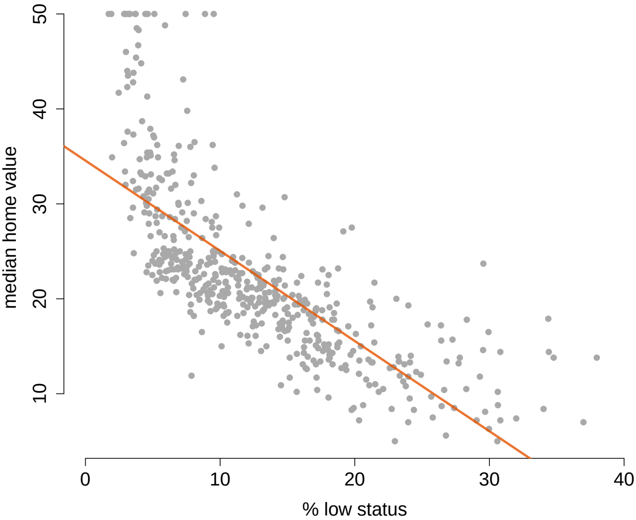
Boston housing data

Prediction at % low status = 30?



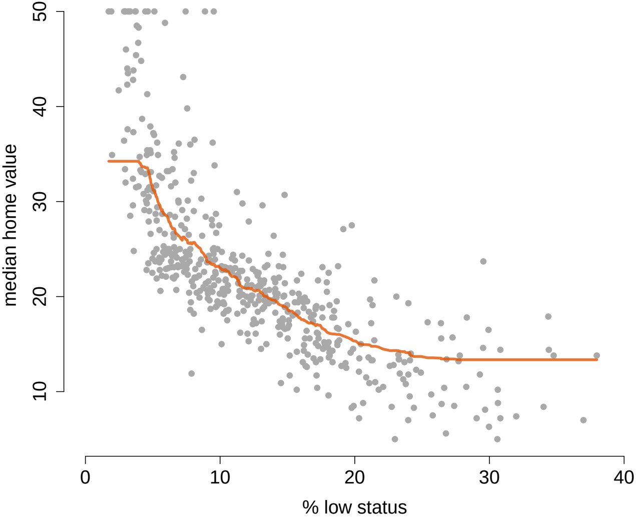
Boston housing data

Prediction at % low status $= 30$?



Boston housing data

Prediction at % low status = 30?



How do we estimate $f(\cdot)$?

1. Choose set of training data: $(Y_1, X_1), \ldots, (Y_N, X_N)$.

How do we estimate $f(\cdot)$?

1. Choose set of training data: $(Y_1, X_1), \ldots, (Y_N, X_N)$.

2. Fit $f(\cdot)$ to training data using:

   - Parametric model, or
   - Nonparametric model

How do we estimate $f(\cdot)$?

1. Choose set of training data: $(Y_1, X_1), \ldots, (Y_N, X_N)$.

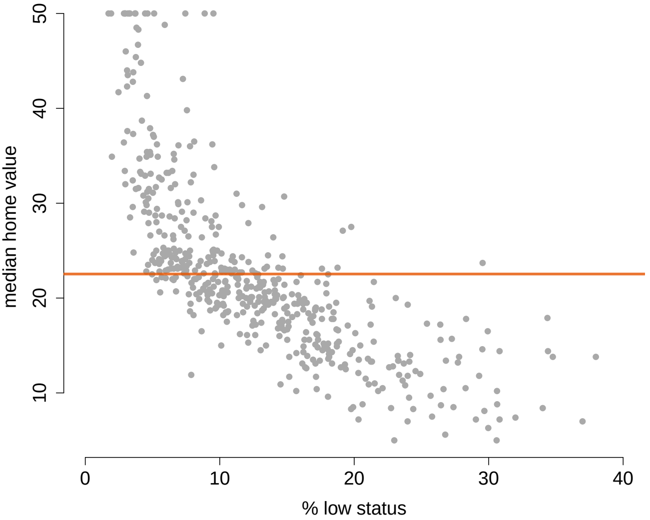
2. Fit $f(\cdot)$ to training data using:

   - Parametric model, or
   - Nonparametric model

3. Evaluate performance on testing data and *adjust*.

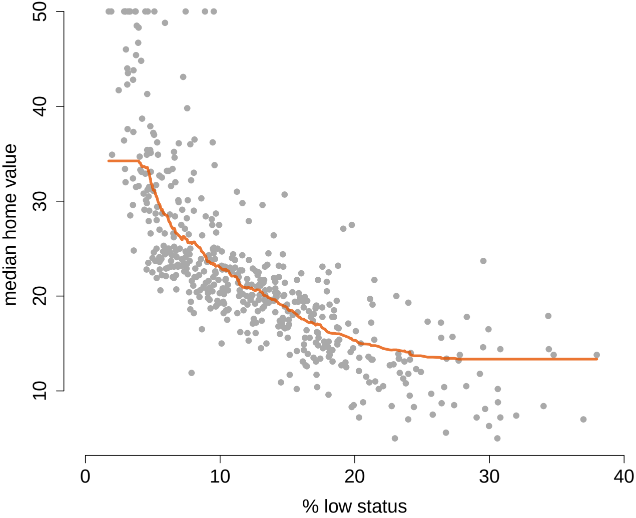
How do we estimate $f(\cdot)$?

Parametric: $Y = \mu + \epsilon$. restrictive assumptions, but simple interpretation.



How do we estimate $f(\cdot)$?

Nonparametric: "Knn" with $k = 100$. flexible assumptions, but complex interpretation.



The challenge when estimating predictions $\widehat{f(\cdot)}$

Balancing *restrictiveness* of assumptions with simplicity of *interpretation*.

Let's look at k-nearest-neighbors (knn)

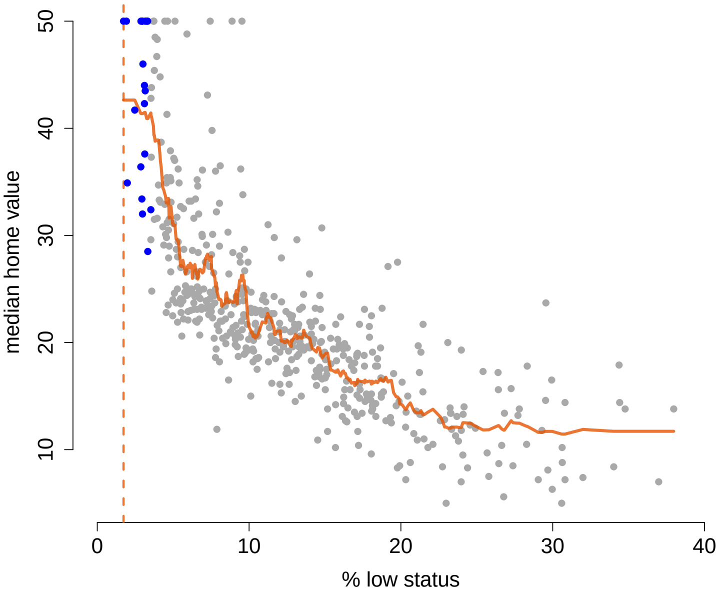
- Prediction at point x, $\widehat{f(x)}$ = average of k nearest points around x.

Let's look at k-nearest-neighbors (knn)

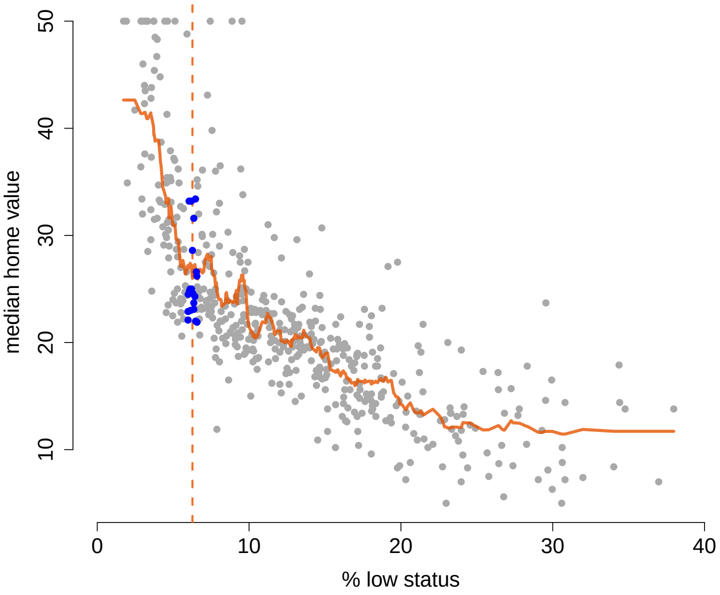
- Prediction at point x, $\widehat{f(x)}$ = average of k nearest points around x.

- Let's look at $k = 20$ ...

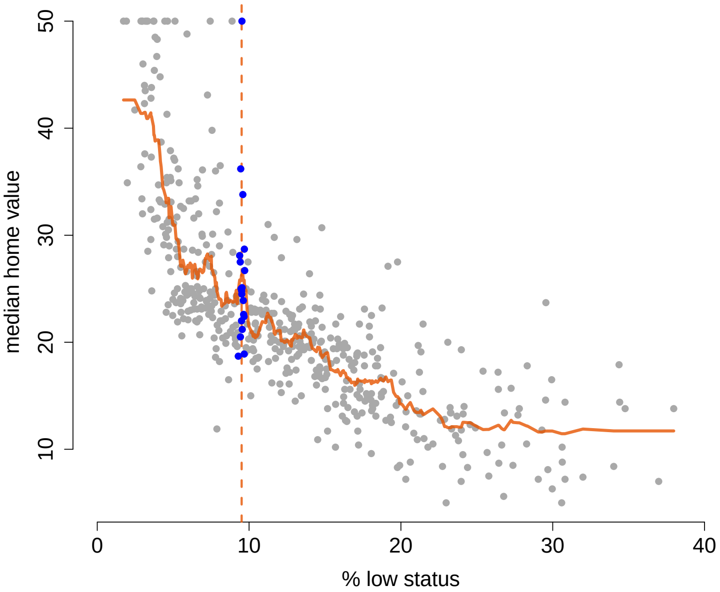
knn with $k = 20$



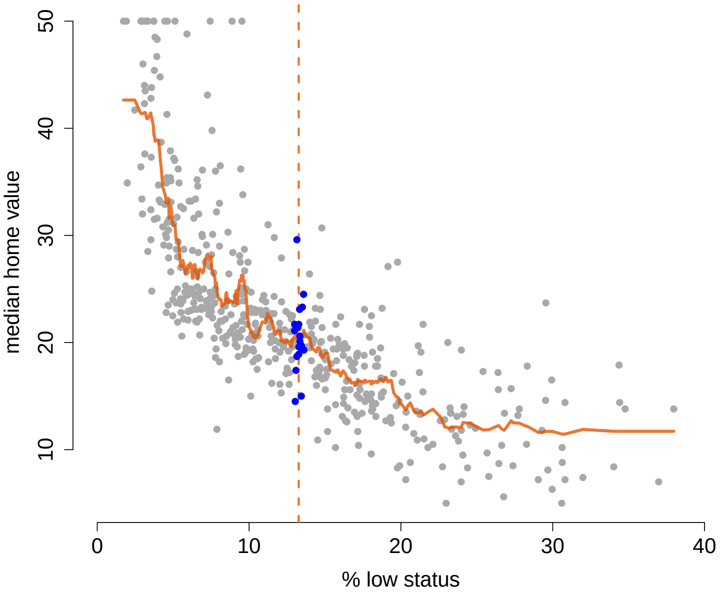
knn with $k = 20$



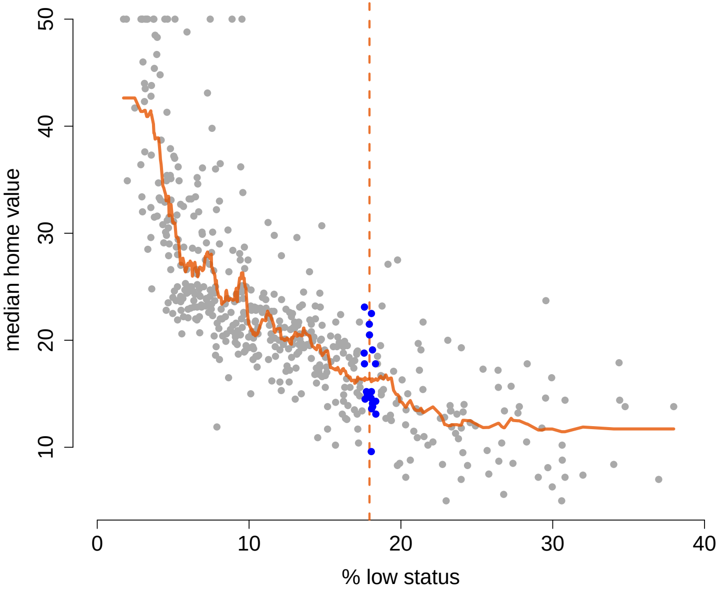
knn with $k = 20$



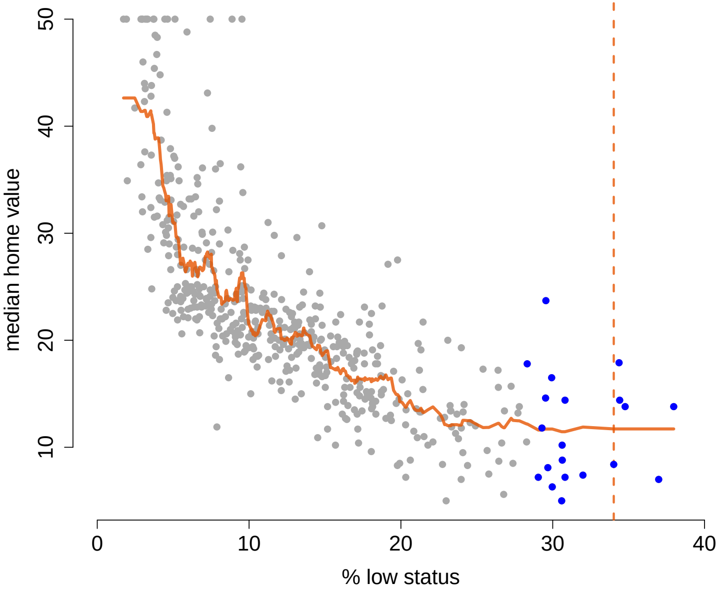
knn with $k = 20$



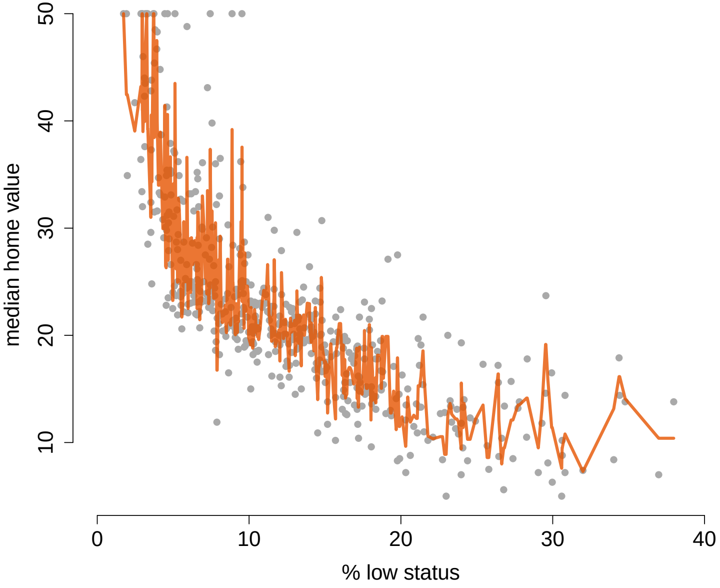
knn with $k = 20$



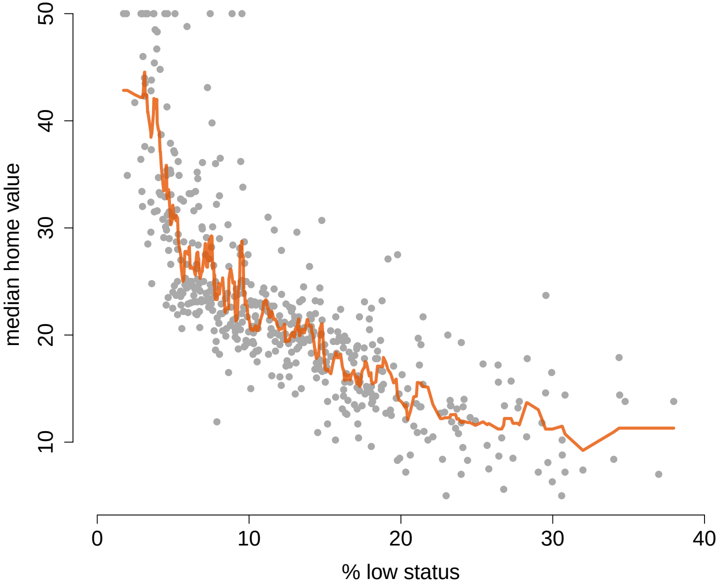
knn with $k = 20$



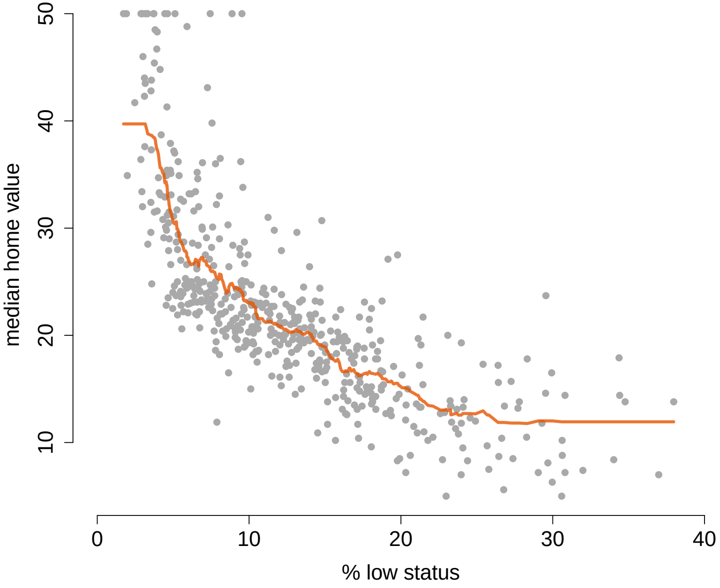
Why don't I choose $k = 2$ instead?



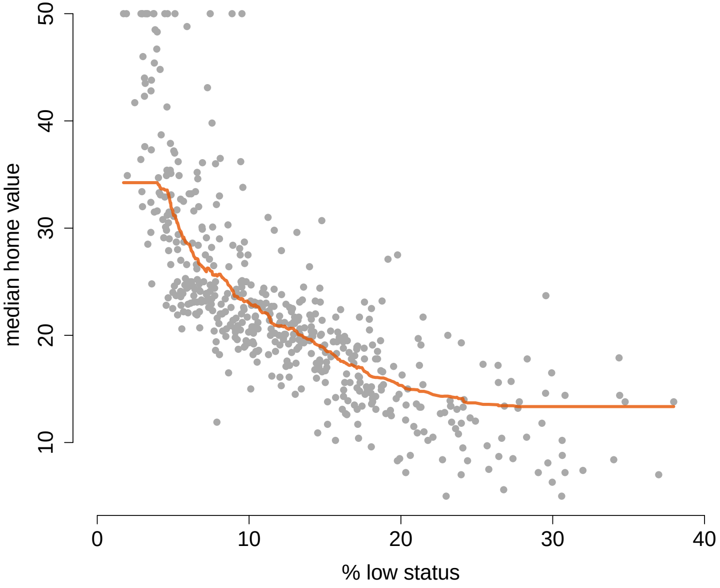
or $k = 10$ ...



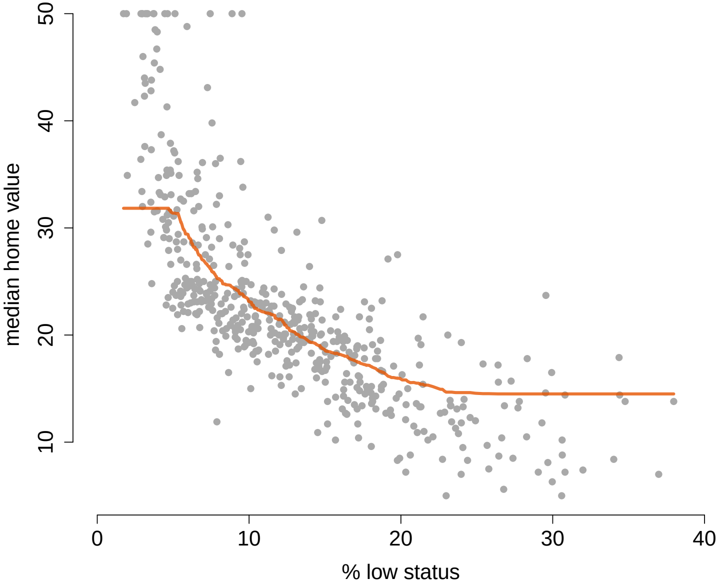
or $k = 50$ ...



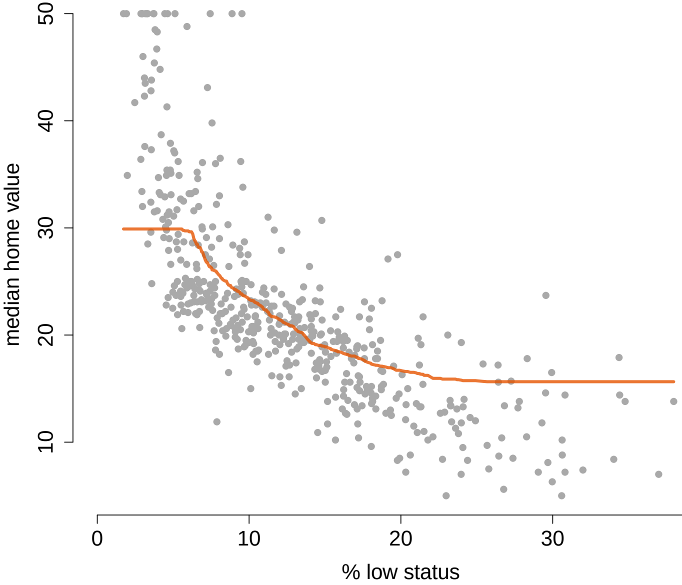
or $k = 100$ ...



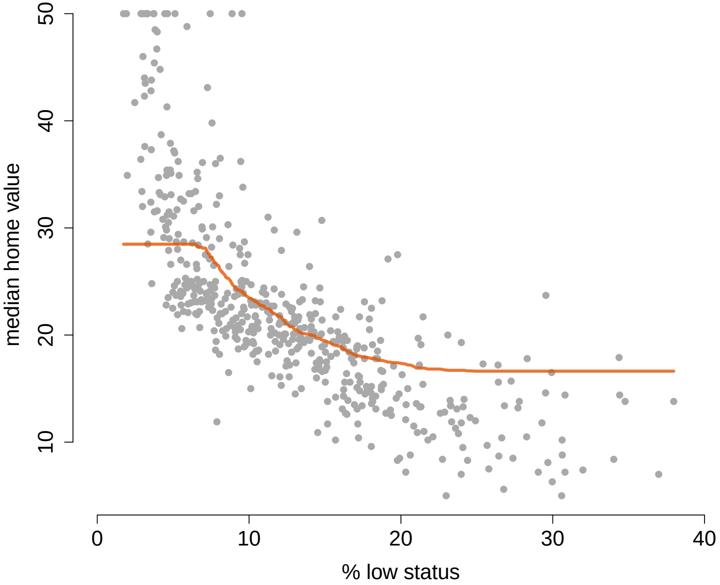
or $k = 150$ ...



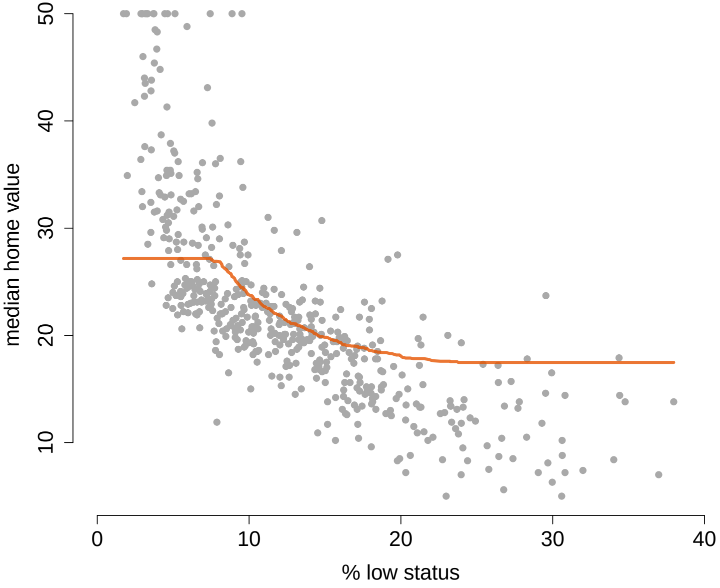
or $k = 200$ ...



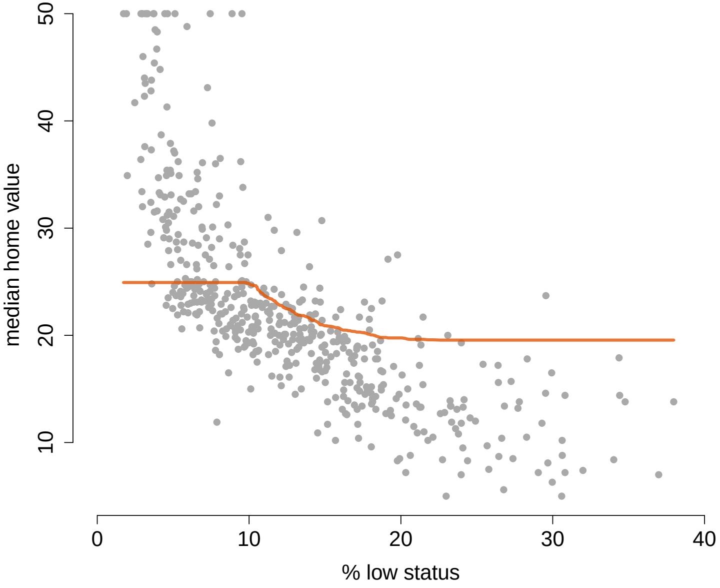
or $k = 250$ ...



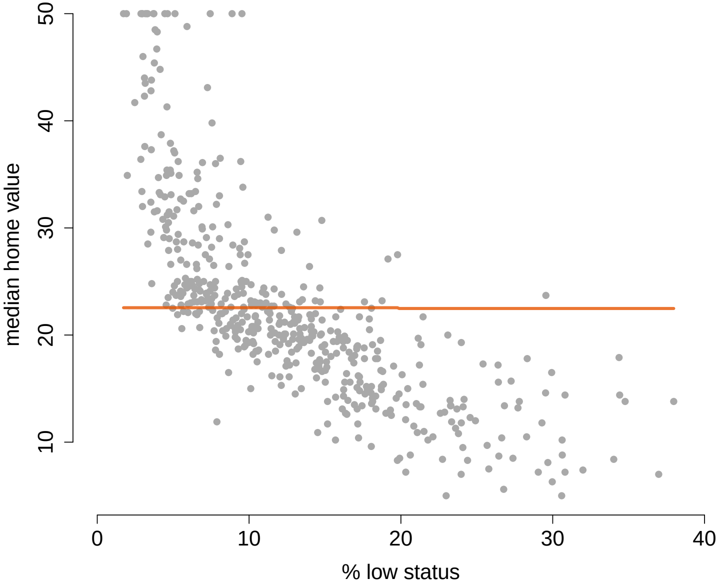
Or $k = 300$ ...



or $k = 400$ ...



or $k = 505$ ...



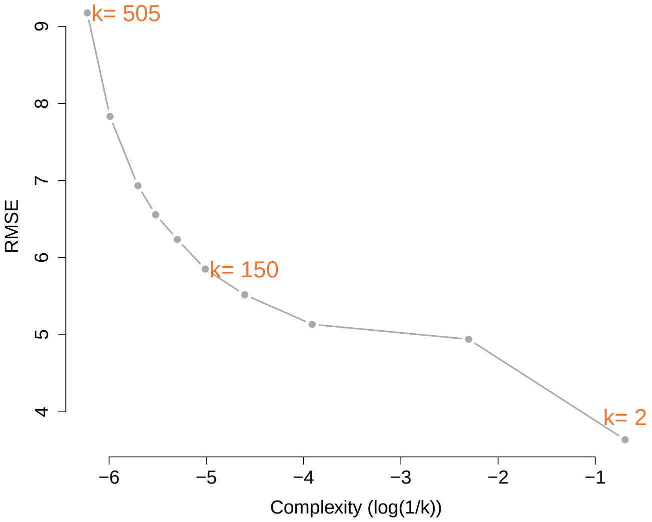
A rigorous way to select

- The root mean squared error measures how accurate my predictions are, on average.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[Y_i - \widehat{f(X_i)}\right]^2}$$

In sample RMSE

It looks like $k = 2$ is the best. Should we choose this model?



We care about out of sample performance

- Suppose we have m additional observations (X_i^o, Y_i^o), for $i = 1, \ldots, m$, that we did not use to fit the model. Let's call this dataset the *validation set* (a.k.a *hold-out set* or *test set*)

We care about out of sample performance

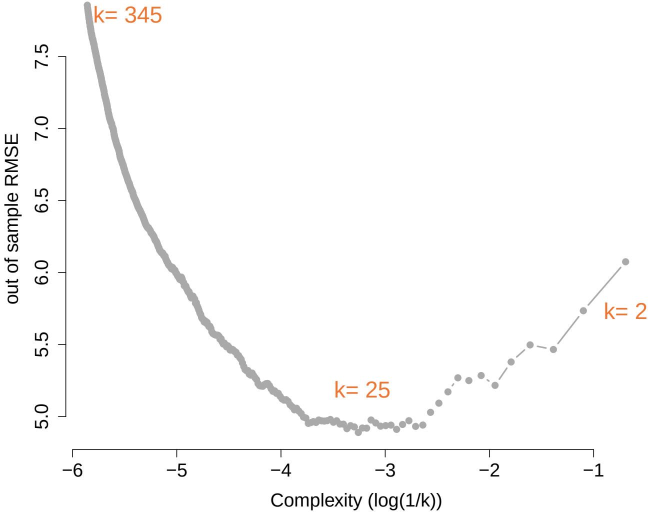
- Suppose we have m additional observations (X_i^o, Y_i^o), for $i = 1, \ldots, m$, that we did not use to fit the model. Let's call this dataset the *validation set* (a.k.a *hold-out set* or *test set*)

- We evaluate the fit with out of sample RMSE:

$$RMSE^o = \sqrt{\frac{1}{m} \sum_{i=1}^{m} \left[Y_i^o - \widehat{f(X_i^o)} \right]^2}$$

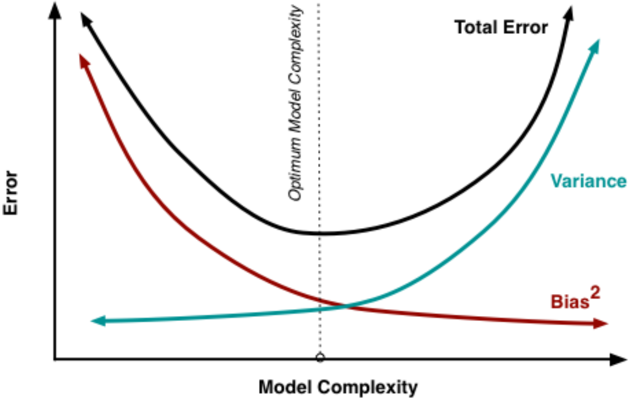
Out of sample RMSE

Fit each model on training set of size 400. Test each model (*out of sample*) on testing set of size 106. Here, we plot the out of sample performance.



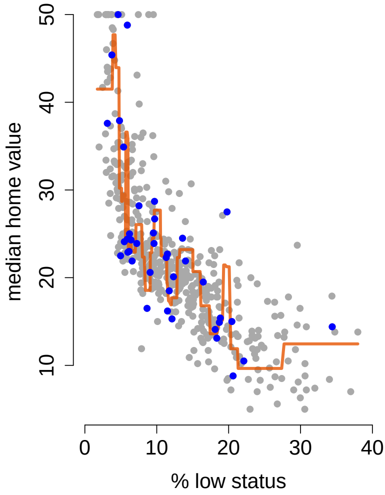
The Bias-variance tradeoff!

When fitting a predictive model, there is a tradeoff between bias and variance of predictions.



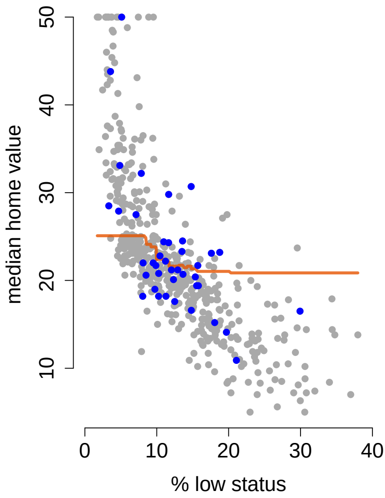
$k = 2$: low bias, high variance

Training set of size 40.



$k = 25$: high bias, low variance

Training set of size 40.

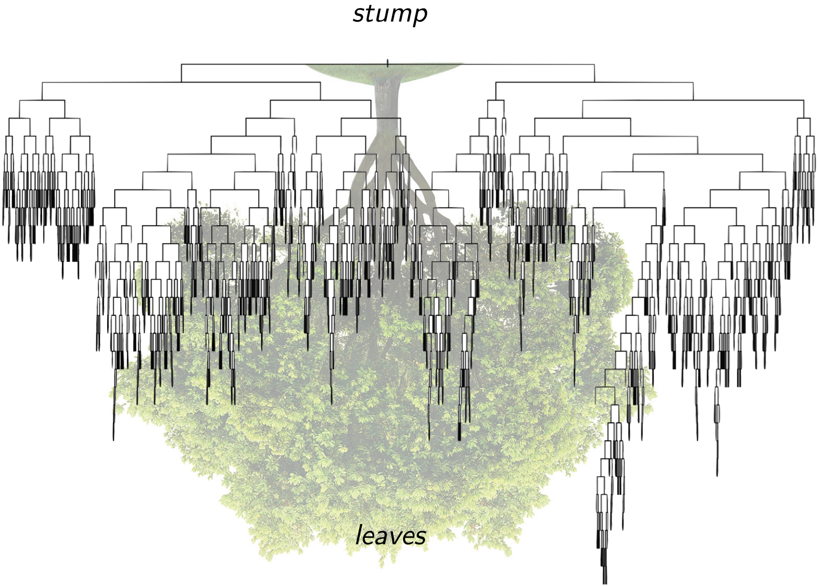


Trees

Trees

Trees

Trees



Trees

- Trees are an intuitive way to estimate $f(\cdot)$.

Trees

- Trees are an intuitive way to estimate $f(\cdot)$.

- Groups of trees (ensembles) is the best machinery for prediction currently available.

Trees

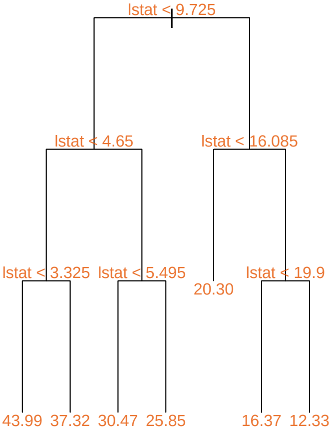
- Trees are an intuitive way to estimate $f(\cdot)$.

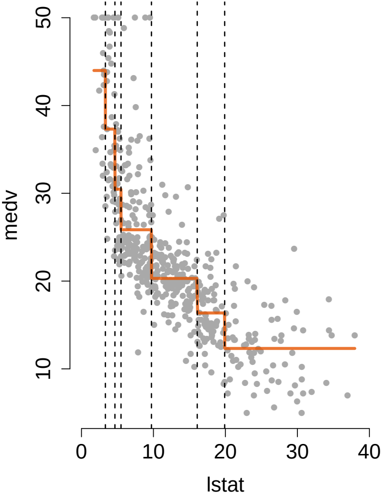
- Groups of trees (ensembles) is the best machinery for prediction currently available.

  1. Bagging
  2. Random Forests
  3. Boosting

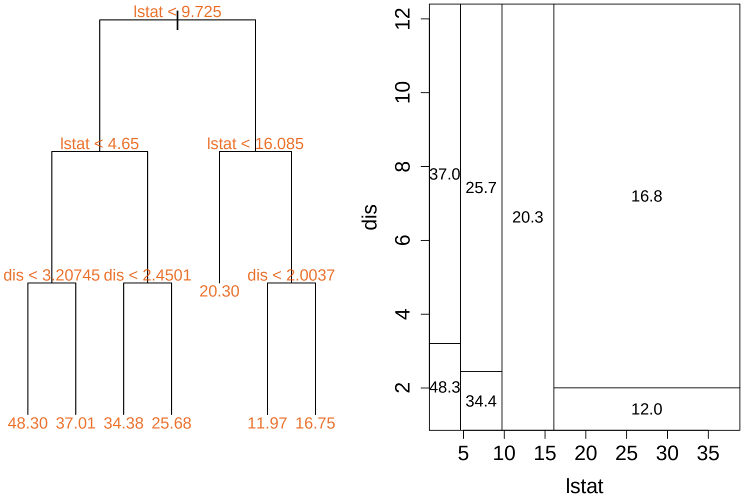
Basic structure - regression trees

Follow rules down tree to come up with prediction. The resulting $f(\cdot)$ is a *step function*! Each region is *average* of training data.





Trees with 2 explanatory variables

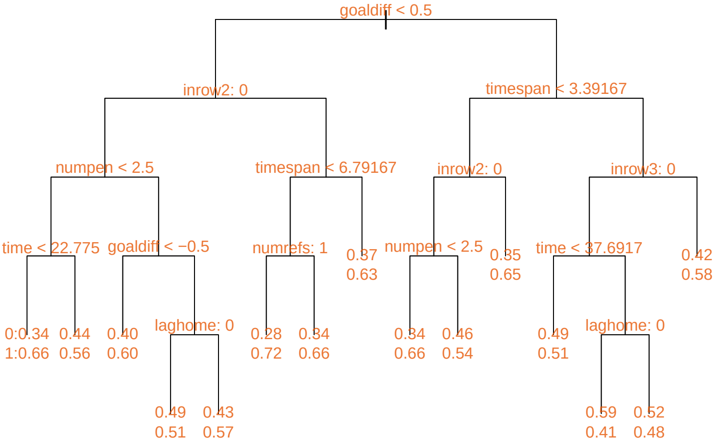


Comparing trees to knn

- Similarities: Both make predictions by *averaging* subsets of X.
- Differences: Trees are more flexible in that *any* subset of the X space can be considered. (knn only considers nearest neighbors).

Basic structure - classification trees

- Y can also be binary. Here, it is whether or not the next penalty in a hockey game is on the other team.
- The leaves are the fraction of training data in the two outcome categories.



Fitting trees and the bias-variance tradeoff

- A big tree is a complex tree.
- We measure a tree's complexity by it's number of leaves.
- As before, we must balance *bias* and *variance* in our predictions when fitting.

Fitting trees and the bias-variance tradeoff

To fit a tree, we try to minimize:

$$C(T, y) = L(T, y) + \alpha\,|T|$$

where,

- $L(T, y)$ is our loss in fitting data y with tree T.
- $|T|$ is the number of bottom nodes in tree T.

For numeric y our loss is usually sum of squared errors, for categorical y we can use the miss-classification rate.

Fitting trees and the bias-variance tradeoff

To fit a tree, we try to minimize:

$$C(T, y) = L(T, y) + \alpha\,|T|$$

where,

- $L(T, y)$ is our loss in fitting data y with tree T.
- $|T|$ is the number of bottom nodes in tree T.

For numeric y our loss is usually sum of squared errors, for categorical y we can use the miss-classification rate.

Note 1: We choose α by cross-validation. Analogous to k in knn.

Fitting trees and the bias-variance tradeoff

To fit a tree, we try to minimize:

$$C(T, y) = L(T, y) + \alpha |T|$$

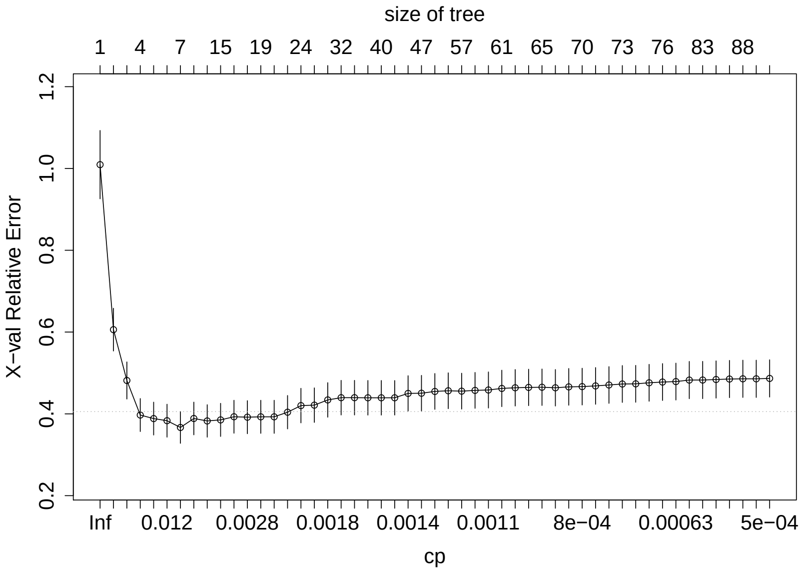
where,

- $L(T, y)$ is our loss in fitting data y with tree T.
- $|T|$ is the number of bottom nodes in tree T.

For numeric y our loss is usually sum of squared errors, for categorical y we can use the miss-classification rate.

Note 1: We choose α by cross-validation. Analogous to k in knn.
Note 2: There are software package that can do this for us!!!

An example of a solution path



Ensembles of trees

Many trees working together do much better than one!

Ensembles of trees

Many trees working together do much better than one!

1. Bagging: Fit many large trees to random (bootstrapped) samples of the data. Average over trees for prediction.

Ensembles of trees

Many trees working together do much better than one!

1. Bagging: Fit many large trees to random (bootstrapped) samples of the data. Average over trees for prediction.

2. Random Forests: Fit many large trees to random (bootstrapped) samples of the data *and* the variables. Average over trees for prediction.

Ensembles of trees

Many trees working together do much better than one!

1. Bagging: Fit many large trees to random (bootstrapped) samples of the data. Average over trees for prediction.

2. Random Forests: Fit many large trees to random (bootstrapped) samples of the data *and* the variables. Average over trees for prediction.

3. Boosting: Fit one tree and "crush" so its small. Then, iteratively fit trees to the residuals of the small tree's fit (called a weak learner). This is like backfitting.

Application to finance data

Predicting firm bankruptcy; Campbell et. al. (JF 2011)

Can we predict when a firm will fail?

- Y_{it}: Is firm i bankrupt in year t?
- $\mathbf{X}_{it-1}$: market and accounting measures observed for firm i in year t, including:
  1. net income to total assets
  2. total liabilities to total assets
  3. annual excess return
  4. standard deviation of return
  5. market cap

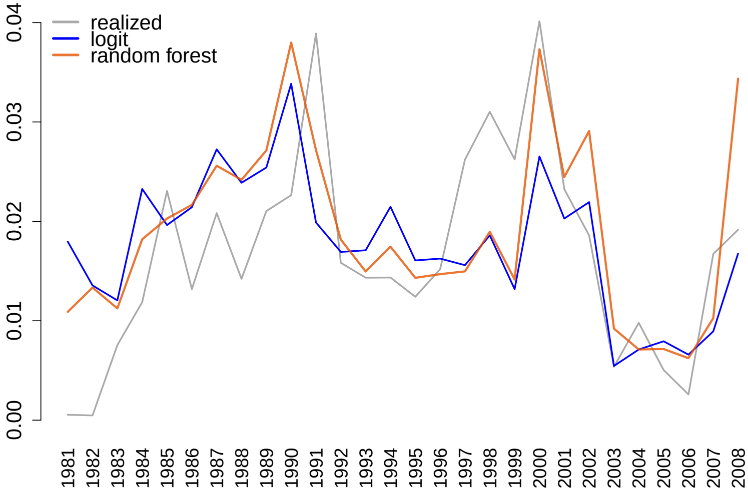
Predicting firm bankruptcy; Campbell et. al. (JF 2011)

- 10390 firms.
- Annual data from 1981-2008.

Let's fit classification trees to this data!

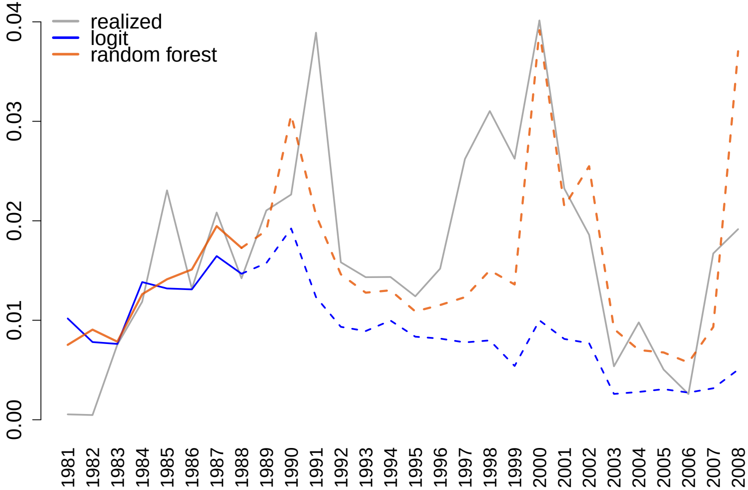
Failure rate each year - in sample

In sample fits for logistic and random forest models.



Failure rate each year - out of sample

Train on 1981-1988, Test on 1989-2008. Random forest dominates the logit!



Future work

This is just the beginning ...

- Build a "distressed" factor based on random forest out of sample prediction.
- Analyze variable importance measures provided by classification trees.
- "Out of sample" analyses become much easier.