Nonparametric Approaches to Regression

• In traditional nonparametric regression, we assume very little about the functional form of the mean response function.

• In particular, we assume the model

\[ m(x_i) \text{ is unknown but is typically assumed to be a smooth, continuous function.} \]

• The \( \varepsilon_i \) are independent r.v.’s from some continuous distribution, with mean zero and variance \( \sigma^2 \).

**Goal:** Estimate the mean response function \( m(x) \).

**Advantages of nonparametric regression:**

• Ideal for situations when we have no prior idea of the relationship between \( Y \) and \( X \).
• By not specifying a parametric form for \( m(x) \), we allow much more flexibility in our model.
• Our model can more easily account for unusual behavior in the data:

• Not as prone to bias in the mean response estimate resulting from choosing the wrong model form.
Disadvantages of nonparametric regression:
• Not as easy to interpret.
• No easy way to describe the relationship between $Y$ and $X$ with a formula written on paper (this must be done with a graph).

Note: Nonparametric regression is sometimes called scatterplot smoothing.
• Specific nonparametric regression techniques are often called smoothers.

Kernel Regression Estimates

• The idea behind kernel regression is to estimate $m(x)$ at each value $x^*$ along the horizontal axis.

• At each value $x^*$, the estimate is simply an

• Consider a “window” of points centered at $x^*$:

• The width of this window is called the ____________.
At each different \( x^* \), the window of points _________ to the left or right

Better idea: Use

This can be done using a ______________ function known as a kernel.

Then, for any \( x^* \),

where the weights

\( K (\cdot) \) is a kernel function, which typically is a density function symmetric about 0.

\( \lambda = \) bandwidth, which controls the smoothness of the estimate of \( m(x) \).

Possible choices of kernel:

Pictures:
Note: The Nadaraya-Watson estimator

is a modification that assures that the weights for the $Y_i$’s will sum to one.

• The choice of bandwidth $\lambda$ is of more practical importance than the choice of kernel.

• The bandwidth controls how many data values are used to compute $m(x^*)$ at each $x^*$.

Large $\lambda \rightarrow$

Small $\lambda \rightarrow$

• Choosing $\lambda$ too large results in an estimate that _____________ the true nature of the relationship between $Y$ and $X$.

• Choosing $\lambda$ too small results in an estimate that follows the “noise” in the data too closely.

• Often the best choice of $\lambda$ is made through visual inspection (pick the roughest estimate that does not fluctuate implausibly?).
• Automatic bandwidth selection methods such as cross-validation are also available – this chooses the $\lambda$ that minimizes a mean squared prediction error:

Example on computer: The R function `ksmooth` performs kernel regression (see web page for examples with various kernel functions and bandwidths).

**Spline Methods**

• A spline is a piecewise polynomial function joined smoothly and continuously at $x$-locations called knots.

• A popular choice to approximate a mean function $m(x)$ is a cubic regression spline.

• This is a piecewise cubic function whose segments’ values and first derivatives are equal at the knot locations.
• This results in a visually smooth-looking overall function.

• The choice of the number of knots determines the smoothness of the resulting estimate:

  Few knots $\rightarrow$

  Many knots $\rightarrow$
• We could place more knots in locations where we expect \( m(x) \) to be wiggly and fewer knots in locations where we expect \( m(x) \) to be quite smooth.

• The estimation of the coefficients of the cubic functions is done through least squares.

• See R examples on simulated data and Old Faithful data, which implement cubic B-splines, a computationally efficient approach to spline estimation.

• A smoothing spline is a cubic spline with a knot at each observed \( x_i \) location.
• The coefficients of the cubic functions are chosen to minimize the penalized SSE:

\[
\lambda \text{ is a smoothing parameter that determines the overall smoothness of the estimate.}
\]

• As \( \lambda \to 0 \), a wiggly estimate is penalized \_______\ and the estimated curve

• As \( \lambda \to \infty \), a wiggly estimate is penalized \_________________________\ and the estimated curve

• See R examples on simulated data and Old Faithful data.

• Inference within nonparametric regression is still being developed, but often it involves bootstrap-type methods.
Regression Trees and Random Forests

- Trees and random forests are other modern, computationally intensive methods for regression.

- **Regression trees** are used when we have one response variable which we want to predict/explain using possibly several explanatory variables.

- The goals of the regression tree approach are the same as the goals of multiple regression:
  1. Determine which explanatory variables have a significant effect on the response.
  2. Predict a value of the response variable corresponding to specified values of the explanatory variables.

- The regression tree is a method that is more algorithm-based than model-based.

- We form a regression tree by considering possible partitions of the data into \( r \) regions based on the value of one of the predictors:

**Example:**

- Calculate the mean of the responses in each region,

- Compute the sum of squared errors (SSE) for this partitioning:
• Of all possible ways to split the data (splitting on any predictor variables and using any splitting boundary), pick the partitioning that produces the smallest SSE.

• Continue the algorithm by making subpartitions based on the most recent partitioning.
• The result is a treelike structure subdividing the data.

• This also works well when a predictor is categorical -- we can subdivide the data based on the categories of the predictor.

• Splitting on one variable separately within partitions of another variable is essentially finding an interaction between the two variables.

• The usual regression diagnostics can be used -- if problems appear, we can try transforming the response (not the predictors).

• Eventually we will want to stop splitting and obtain our final tree.
• Once we obtain our final tree, we can predict the response for any observation (either in our sample, or a new observation) by following the splits (based on the observation’s predictor values) until we reach a “terminal node” of the tree.

• The predicted response value is the mean response of all the sampled observations corresponding to that terminal node.

• A criterion to select the “best” tree is the cost-complexity:

• The first piece measures fit and the second piece penalizes an overly complex tree.
• Another approach to tree selection is **cross-validation.**

• We select a random subset of the data, build a tree with that subset, and use the tree to predict the responses of the remaining data.

• Then a cross-validation prediction error can be calculated: A tree with low CV error (as measured by MSPR) is preferred.

• The `rpart` function in the `rpart` package of R produces regression tree analyses.

• More (or less) complex trees may be obtained by adjusting the `cp` argument in the `prune.rpart` function.

• The `cp` value is directly proportional to \( \lambda \), so a larger value of `cp` encourages a ___________ tree.

• The `plotcp` function can guide tree selection by plotting CV error against `cp`: We look for the elbow in the plot.

**Examples** (Boston housing data, University admissions data): A plot of the graph of the tree reveals the important variables.

• Classification Trees work similarly and are used when the response is categorical.
Random Forests

- The random forest approach is an **ensemble** method -- it generates many individual predictions and aggregates them to produce a better overall method.

- As the name suggests, a random forest consists of **many trees**.

- It relies on the principle of **bagging** (bootstrap aggregating) proposed by Leo Breiman.

- Different trees are constructed using $n_{\text{tree}}$ bootstrap resamples of the data, and the nodes are split based on random subsets of predictors, each of size $m_{\text{try}}$.

- In regression, prediction is done by averaging predicted response values across the predicted trees.

- The error rate is typically assessed by predicting out-of-bag (OOB) data -- the data not chosen for the bootstrap sample -- using each constructed tree.

- The `randomForest` function in the `randomForest` package in R will obtain a random forest, for either regression (continuous response) or classification (categorical response).

- It also provides a measure of which explanatory variables are most important.

- See examples on the course web page.