UNIVERSITY of SOUTH FLORIDA

Nonlinear Models
(Support Vector Regression, K-Nearest Neighbors)

Spring 2020

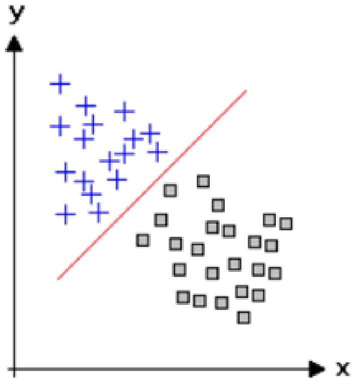
Instructor: Ankit Shah, Ph.D.

Types of Predictive Methods

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} {x_1}^2 + \beta_{22} {x_2}^2$$

$$y = e^{\beta x_1}$$



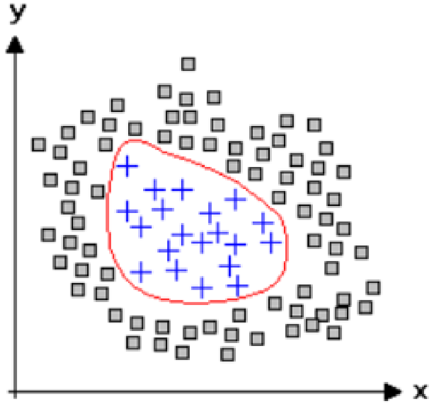
Predictive Methods

Linear

Linear Regression
Logistic Regression
Ridge Regression
Least Absolute Shrinkage and Selection Operator (LASSO)

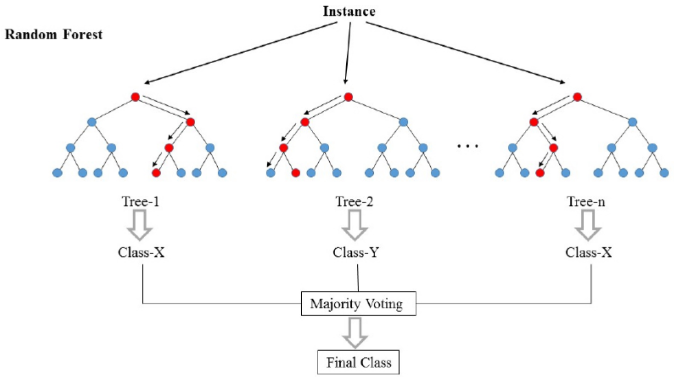
Nonlinear

Support Vector Machine/Regression
Neural Networks



Tree-based

Classification and Regression Trees
Random Forests



Linear models are linear in the parameters that need to be estimated, but not necessarily in the independent variables

Regression

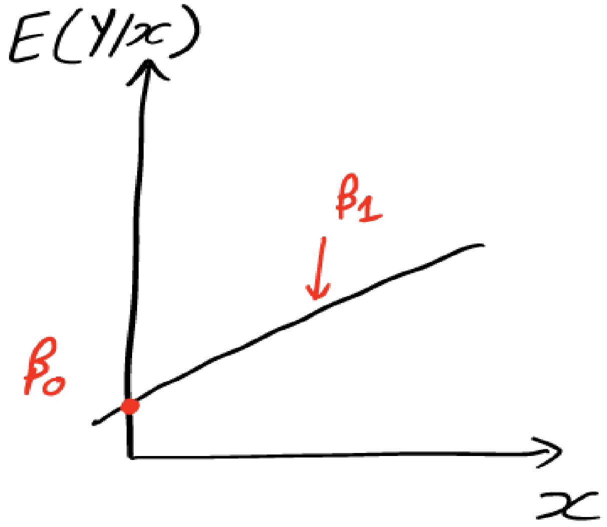
Simple Linear Regression

- Simple Linear Regression:
  - One predictor variable
  - First-order linear model

$$E(Y_i|x_i) = \beta_0 + \beta_1 x_i$$

where x_i represents the value of the predictor in the ith sample, β_0 is the intercept and β_1 is the slope

x	y



- Intercept: expected response value when x takes the value of 0
- Slope of the line: change in the expected response value for one unit change in x

- If we know the values of the parameters β_0 and β_1, then we can predict y_i with good accuracy
- However, if we do not know these values, then we need to estimate $\widehat{\beta_0}$, $\widehat{\beta_1}$
- Once we obtain these estimates, then we can find $\hat{y}_i = \widehat{\beta_0} + \widehat{\beta_1} x_i$

Multiple Linear Regression

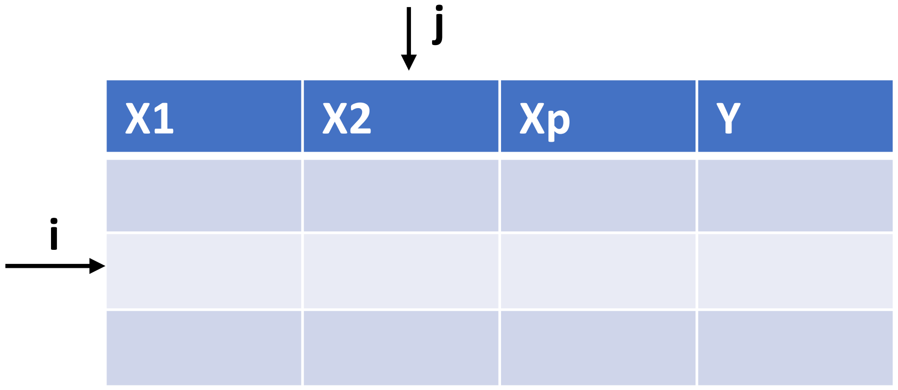
- The simple linear regression model can be extended to allow for more than one predictors:

$$E(Y_i|x_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \dots + \beta_p x_{ip}$$

x_i represents the vector of values for p different predictor variables for the ith sample

x_{ij} is the value of the jth predictor variable for the ith case

β_j is the change in the mean of the response variable due to a one unit increase in x_{ij}, when the other predictor variable values are held fixed



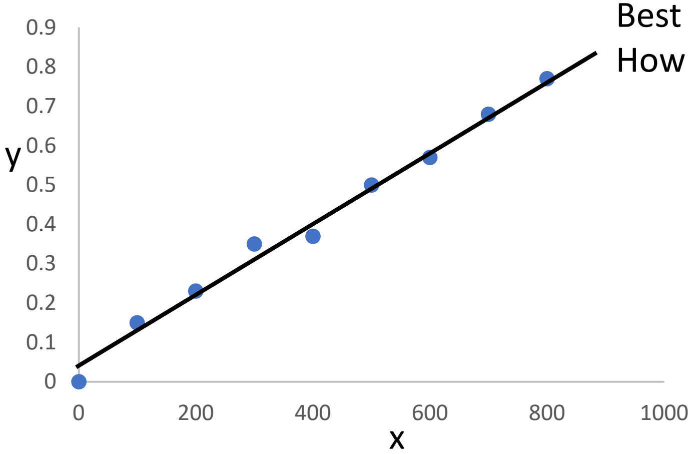
X1	X2	Xp	Y

- Ordinary Least Squares method for fitting a simple linear regression model can be extended and the least squares solution can be derived using matrices

Linear Regression

- Find the relationship between the force applied to a beam and the deflection it causes.

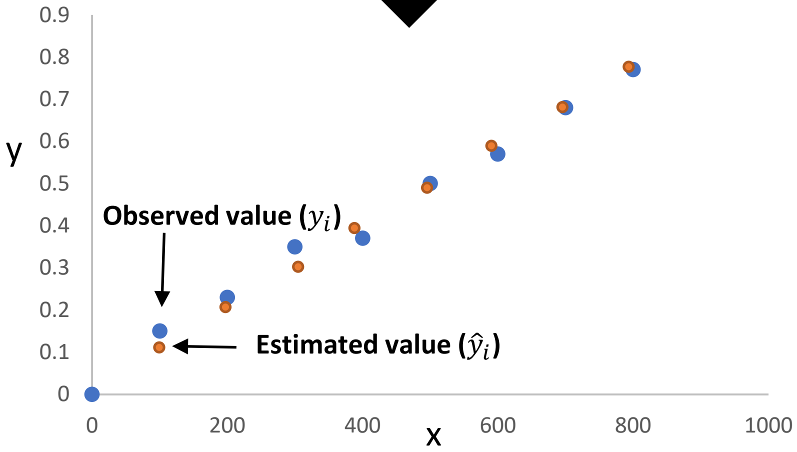
x (force in lbs.)	y (deflection in in.)
0	0
100	0.15
200	0.23
300	0.35
400	0.37
500	0.5
600	0.57
700	0.68
800	0.77



Best Fit Line

How do we find such a line?

Least Squares Fit



$\hat{y}_i - y_i$ is called i$^{\text{th}}$ error or i$^{\text{th}}$ residual

Objective:

Minimize $\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$

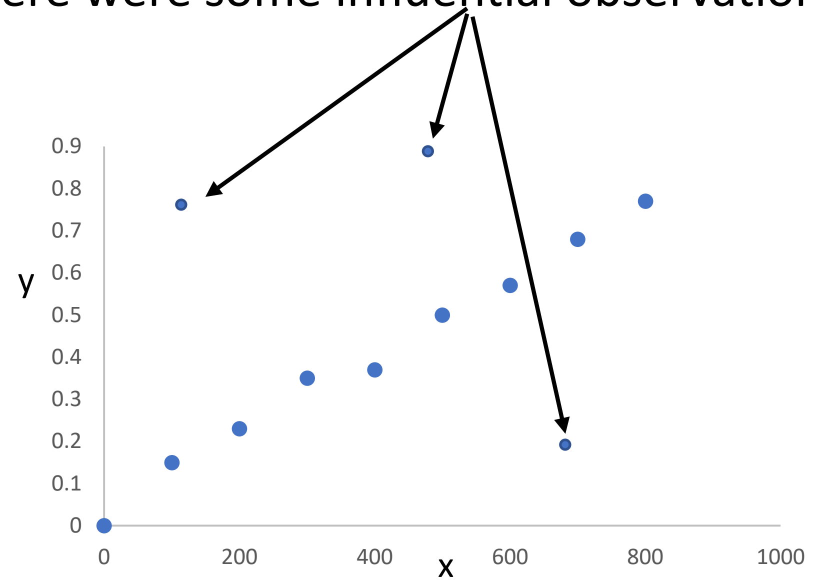
Sum of Squared Residuals (SSR)

or Sum of Squared Errors (SSE)

or **Residual Sum of Squares (RSS)**

Linear Regression

- What will happen if there were some influential observations in your training data set?



Observations that cause significant changes in the parameter estimates are called influential observations

- Linear regression seeks to find estimates that minimize RSS
  - Hence, it will chase the observations that are far away from the overall trend of the majority of the data points

Huber Function

Huber Function

- Linear regression is prone to chasing observations that are away from the overall trend of the majority of the data
  - There are no tuning parameters for multiple regression methods
- One approach to deal with the influential observations is to simply consider taking the absolute residuals

  Minimize $\sum_{i=1}^{n} |y_i - \hat{y}_i|$

- Another approach is to use a robust loss function

$$L(y, \hat{y}) = \begin{cases} (y - \hat{y})^2 & \rightarrow |y - \hat{y}| \leq \alpha \\ |y - \hat{y}| & \rightarrow |y - \hat{y}| > \alpha \end{cases}$$

Huber Function

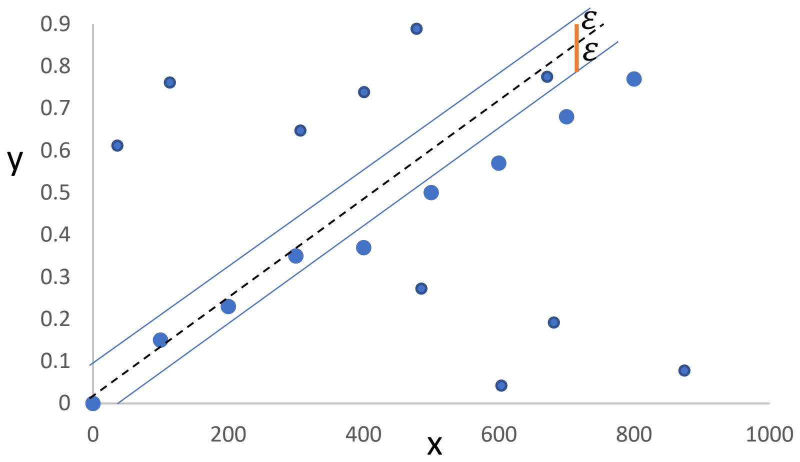
- In other words, Huber Function uses the squared residuals (RSS) when the residuals are small and absolute residuals when the residuals are large
  - Tuning parameter is α

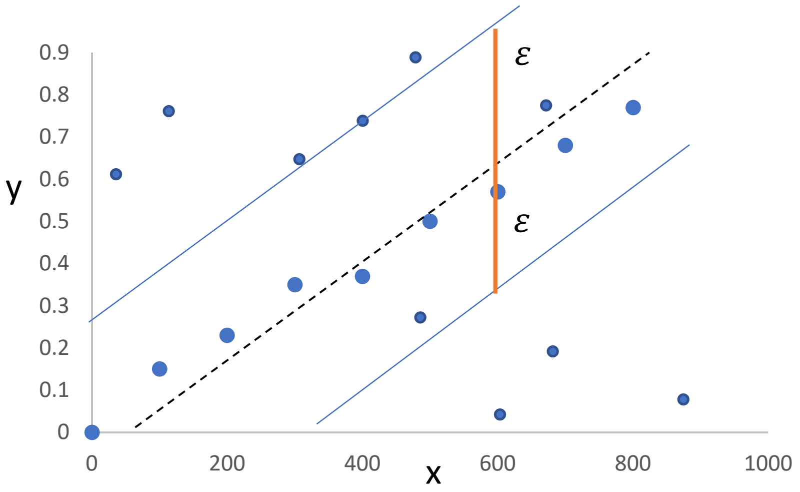
What happens if we keep the value of α very high?

Support Vector Regression

Support Vector Regression (SVR)

- Support Vector Regression is another method to minimize the effect of the influential observations





Support Vector Regression (SVR)

- Support Vector Regression is another method to minimize the effect of the influential observations
- Method:
  - Choose a small value, say, ε
  - Data points with absolute residuals greater than ε contribute to the regression fit
  - Data points whose residuals are small (less than or equal to ε) have no effect on the regression equation
  - Note: this method looks at the absolute residuals and not the squared residuals
    - Softens the impact of the influential observations on the model fit

- The loss function used to estimate the parameters is given as follows:

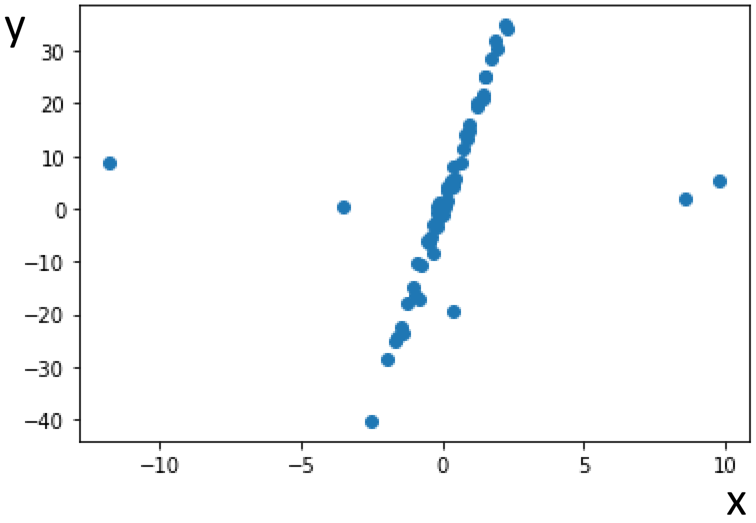
$$\underset{\substack{\nearrow\\ \text{cost penalty}}}{C}\sum_{i=1}^{n} \underset{\substack{\uparrow\\ \varepsilon\text{-sensitive function}}}{L_{\epsilon}}\,(y_i - \hat{y}_i) + \sum_{j=1}^{\mathrm{p}} \beta_j^2$$

If $(y_i - \hat{y}_i) \leq \varepsilon$, then 0

Important:
The penalty is attached to the residuals

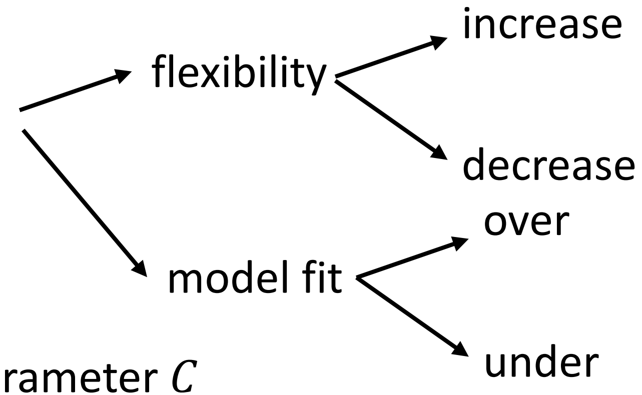
Support Vector Regression (SVR)

- Suppose we obtained the following scatter plot between x and y for a given training data set



SVR Loss Function:

$$C \sum_{i=1}^{n} L_\epsilon \ (y_i - \hat{y}_i) + \sum_{j=1}^{p} \beta_j^2$$



What happens if we assign a large cost penalty?

- In practice, we may want to fix the value of ε and then tune the cost penalty parameter C

- Import SVR from sklearn.svm module

```
from sklearn.svm import SVR
```

- Call the SVR function

```
model_SVR = SVR(C=1.0, epsilon=0.1, kernel = ‘linear’)
```

inversely proportional

Support Vector Regression (SVR)

- Given a new sample, u, we can find the predicted value according to the equation:

$$\beta_0 + \beta_1 u_1 + \beta_2 u_2 + \ \ldots + \beta_\mathrm{p} u_p = 0$$

This can be further expressed as:

- $\beta_0 + \sum_{j=1}^{\mathrm{p}} \beta_\mathrm{j}\ u_j$

obtained from the training data points

- $\beta_0 + \sum_{j=1}^{\mathrm{p}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_i x_{ij}\ u_j$

dot product

- $\beta_0 + \sum_{i=1}^{\mathrm{n}} \alpha_i \sum_{\mathrm{j}=1}^{\mathrm{p}} x_{ij}\ u_j$

Kernel function

- $\beta_0 + \sum_{i=1}^{\mathrm{n}} \alpha_i\ K(x_i, u)$

- It is to be noted that we need the training data points to calculate the value for the test sample (predictions)
- We now have a set of unknow parameters α_i
  - How many?
  - What happens to the α values for the data points whose residuals are less than or equal to ε?
- Support Vectors are data points whose value for $\alpha \neq 0$
  - In other words, data points whose residuals are greater than ε
- Sum of cross products will greatly affect the model if the predictor scales are different
  - Hence, it is critical to center and scale the predictors

Support Vector Regression (SVR)

- Given a new sample, u, we can find the predicted value according to the equation:

$$\beta_0 + \beta_1 u_1 + \beta_2 u_2 + \ \ldots + \beta_{\mathrm{p}} u_p = 0$$

This can be further expressed as:

- $\beta_0 + \sum_{j=1}^{\mathrm{p}} \beta_{\mathrm{j}} \ u_j$

obtained from the training data points

- $\beta_0 + \sum_{j=1}^{\mathrm{p}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_i x_{ij} \ u_j$

dot product

- $\beta_0 + \sum_{i=1}^{\mathrm{n}} \alpha_i \sum_{\mathrm{j}=1}^{\mathrm{p}} x_{ij} \ u_j$

Kernel function

- $\beta_0 + \sum_{i=1}^{\mathrm{n}} \alpha_i \ K(x_i, u)$

A polynomial kernel of degree d:

$$\mathrm{K}(x_{\mathrm{i}}, u) = \left(\sum_{\mathrm{j}=1}^{\mathrm{p}} x_{ij} \ u_j\right)^{\mathrm{d}}$$

A radial basis kernel:

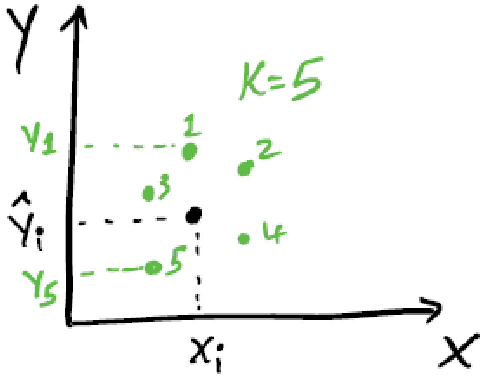
$$\mathrm{K}(x_{\mathrm{i}}, u) = \exp\left(-\gamma \sum_{j=1:p} (x_{\mathrm{ij}} - u_j)^{2}\right)$$

K-Nearest Neighbors Regression

K-Nearest Neighbors (KNN) Regression

- Objective is to find the K nearest data points from the new data point and then predict the value of the response variable based on the K response values
- For regression, this value can be the mean of the K response values

$$\hat{y}_i = \frac{1}{k}\sum_{\mathrm{m=1}}^{k} y_m$$



- How do you find the nearest neighbors?
  - One of the popular distance metrics is the Euclidean distance (straight-line distance between 2 data points)

$$(\sum_{\mathrm{j=1}}^{\mathrm{p}} (x_{aj} - x_{bj})^2)^{1/2}$$

An Example

Training data

y = Health Score	x1 = Height (cm)	x2 =Weight (grams)
1.6	-1.186	-0.707
2.3	-0.074	1.414
3.1	1.260	-0.707

New (Test) Sample

y	x1	x2
?	0.996	1.414

If the predictor data are in different measurements, we must first center and scale the data

$$\left(\sum_{j=1}^{p}(x_{aj}-x_{bj})^{2}\right)^{1/2}$$

	(New Sample - Data Point 1)^2	(New Sample - Data Point 2)^2	(New Sample - Data Point 3)^2
x1	4.76	1.14	0.07
x2	4.50	0.00	4.50
Sum	9.26	1.14	4.57
Sqrt	3.04	1.07	2.14

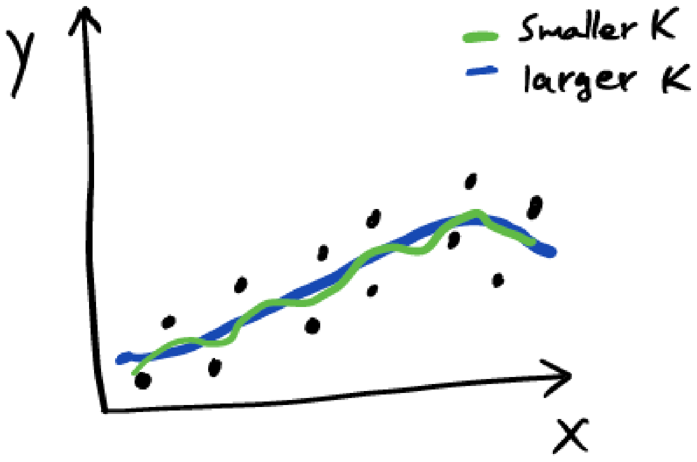
What if the KNN method was weighted?
i.e., 70% weight associated with the nearest neighbor

$$\hat{y}_i = \frac{1}{k}\sum_{m=1}^{k} y_m$$

for K = 2

y	x1	x2
(2.3+3.1)/2 = 2.7	0.996	1.414

KNN Regression



- Smaller value of K will provide more flexibility
  - How does this impact Bias or Variance?
- Larger value of K will reduce flexibility
  - How does this impact Bias or Variance?

- If one or more predictor values for a sample are missing, the distance cannot be calculated
  - Hence, either the sample may have to be removed or values need to be imputed
- Noisy predictors also contribute to poor response value estimates
- Value of K is determined by resampling methods using the training data set
- Method works well when p is not too large, else performs poorly when p is large (curse of dimensionality)
- Efficient data structures such as a k-dimensional tree representation can help with computational challenges
- KNN Regression does not help with interpretability