Non Linear Models

MODELLING NON LINEARITIES in DATA using various Non linear functions-

The most basic idea behind adding Non linear properties in the Model is by transforming the data or the variables,alomst every trick transforms the variables to Model Non linearitites.

Say Kernel Smoothing techniques , Splines or Step functions etc.

```
#Package containing the Dataset
require(ISLR)
attach(Wage)#Dataset
```

Polynomial Regression

First we will use polynomials , and focus on only one predictor age.

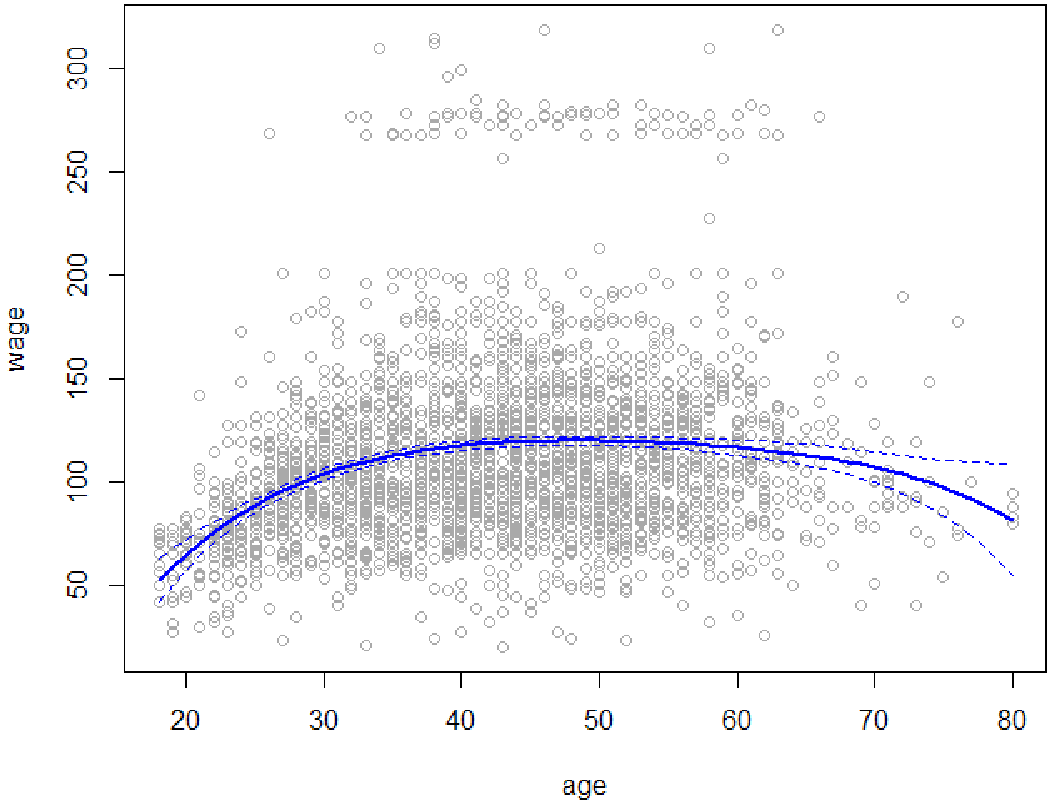
```
mod<-lm(wage~poly(age,4),data =Wage)
#Summary of the Model
summary(mod)

## 
## Call:
## lm(formula = wage ~ poly(age, 4), data = Wage)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -98.707 -24.626  -4.993  15.217 203.693 
## 
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)    
## (Intercept)    111.7036     0.7287 153.283  < 2e-16 ***
## poly(age, 4)1  447.0679    39.9148  11.201  < 2e-16 ***
## poly(age, 4)2 -478.3158    39.9148 -11.983  < 2e-16 ***
## poly(age, 4)3  125.5217    39.9148   3.145  0.00168 ** 
## poly(age, 4)4  -77.9112    39.9148  -1.952  0.05104 .  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504 
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

It looks like the ***Quadatric*** coefficient is not Sifgificant.So we can stop tell 3.

Plotting the Model and Making Predictions-

```
#Range of age variable
agelims<-range(age)
#Generating Test Data
age.grid<-seq(from=agelims[1], to = agelims[2])
#Making Predctions on Test data
pred<-predict(mod,newdata = list(age=age.grid),se=TRUE)
#Standard Error Bands- within 2 Standard Deviations
se.tab<-cbind(pred$fit+2*pred$se.fit,pred$fit -  2*pred$se.fit)
plot(age,wage,col="darkgrey")
#Plotting the age values vs Predicted Wage values for those Ages
lines(age.grid,pred$fit,col="blue",lwd=2)
#To plot the Error bands around the Regression Line
matlines(x=age.grid,y=se.tab,lty =2,col="blue")
```



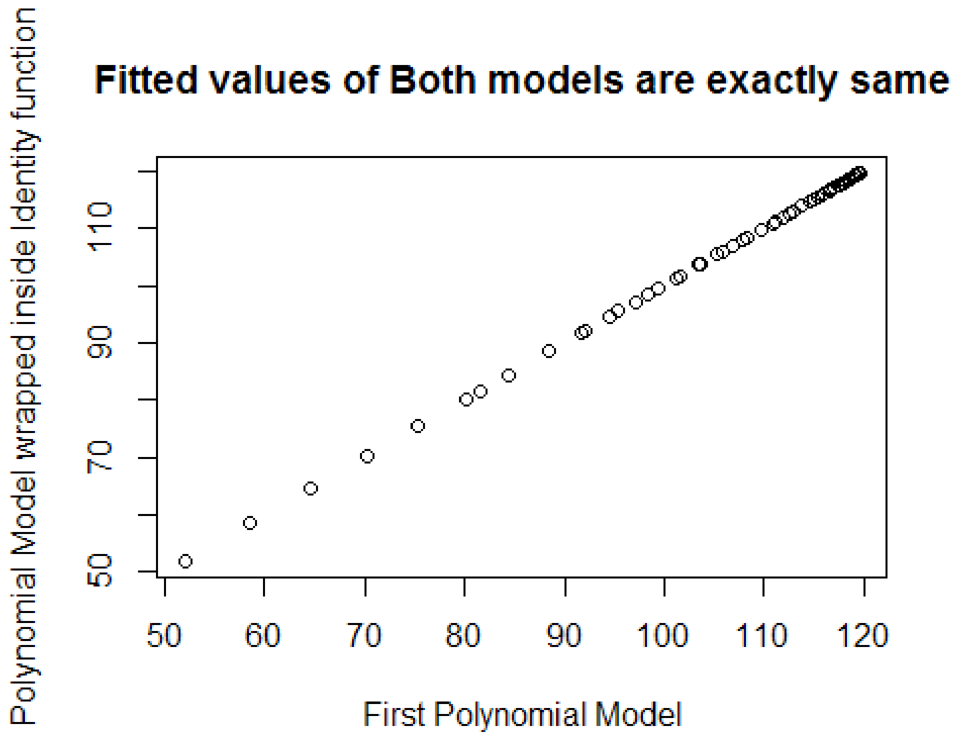
Other Methods to fit polynomials

This time we are going to wrap the polynimials inside the I() Identity function and now we are representing the polynomials on a different basis.

```r
#This time we will use different basis of polynomials
fit2<-lm(wage ~ age + I(age^2) + I(age^3) + I(age^4),data = Wage)
summary(fit2)
```

```
##
## Call:
## lm(formula = wage ~ age + I(age^2) + I(age^3) + I(age^4), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.707 -24.626  -4.993  15.217 203.693
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.842e+02  6.004e+01  -3.067 0.002180 **
## age          2.125e+01  5.887e+00   3.609 0.000312 ***
## I(age^2)    -5.639e-01  2.061e-01  -2.736 0.006261 **
## I(age^3)     6.811e-03  3.066e-03   2.221 0.026398 *
## I(age^4)    -3.204e-05  1.641e-05  -1.952 0.051039 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

```r
plot(fitted(mod),fitted(fit2),xlab="First Polynomial Model",ylab="Polynomial
Model wrapped inside Identity function", main="Fitted values of Both models
are exactly same")
```



We can notice that the coefficients and the summary is different though we have used the same degree of polynomials and this is merely due to the different representations of the polynomils using Identity I() function.

Things we are interested in is the Fitted polynomial and we can notice that the fitted values of both The model above and this Model has not changed.

Now we will use anova() to test different Models with different Predictors

```
#Making Nested Models-i.e Each Next Model includes previous Model and is a
special case for previous one
mod1<-lm(wage ~ education , data = Wage)
mod2<-lm(wage ~ education + age,data = Wage)
mod3<-lm(wage ~ education + age + poly(age,2),data = Wage)
mod4<-lm(wage ~ education + age + poly(age,3),data = Wage)
#using anova() function
anova(mod1,mod2,mod3,mod4)

## Analysis of Variance Table
##
## Model 1: wage ~ education
## Model 2: wage ~ education + age
## Model 3: wage ~ education + age + poly(age, 2)
## Model 4: wage ~ education + age + poly(age, 3)
##   Res.Df     RSS Df Sum of Sq       F Pr(>F)
## 1   2995 3995721
## 2   2994 3867992  1    127729 102.7378 <2e-16 ***
## 3   2993 3725395  1    142597 114.6969 <2e-16 ***
## 4   2992 3719809  1      5587   4.4936 0.0341 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
BIC(mod1,mod2,mod3,mod4)
```

```
##      df      BIC
## mod1  6 30144.77
## mod2  7 30055.31
## mod3  8 29950.63
## mod4  9 29954.13
```

Seeing the Above values,Model 4 which is the most Complex one is the most Insignificant Model as the p-values indicate.Though the RSS value of Model 4 is least,and this is a expected as it fitting data too ***hard(Overfitting)***.

Model2 and Model3 are the best ones and seem to balance the Bias-Variace Tradeoffs.

Polynomial Logistic Regression

```
#Logistic Regression Model the Binary Response variable;
logmod<-glm(I(wage > 250 ) ~ poly(age,3),data = Wage , family = "binomial")
summary(logmod)
```

```
##
## Call:
## glm(formula = I(wage > 250) ~ poly(age, 3), family = "binomial",
##     data = Wage)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2808  -0.2736  -0.2487  -0.1758   3.2868
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -3.8486     0.1597 -24.100  < 2e-16 ***
## poly(age, 3)1   37.8846    11.4818   3.300 0.000968 ***
## poly(age, 3)2 -29.5129    10.5626  -2.794 0.005205 **
## poly(age, 3)3   9.7966     8.9990   1.089 0.276317
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 730.53  on 2999  degrees of freedom
## Residual deviance: 707.92  on 2996  degrees of freedom
## AIC: 715.92
```

```
##
## Number of Fisher Scoring iterations: 8
```

```
#doing Predictions
pred2<-predict(logmod,newdata = list(age=age.grid),se=TRUE)
#Standard Error Bands
#a Matrix with 3 columns
#Confidence intervals
se.band<-pred2$fit + cbind(fit=0,lower=-2*pred2$se.fit , upper =
2*pred2$se.fit )
se.band[1:5,]
```

```
##         fit      lower     upper
## 1 -7.664756 -10.759826 -4.569686
## 2 -7.324776 -10.106699 -4.542852
## 3 -7.001732  -9.492821 -4.510643
## 4 -6.695229  -8.917158 -4.473300
## 5 -6.404868  -8.378691 -4.431045
```

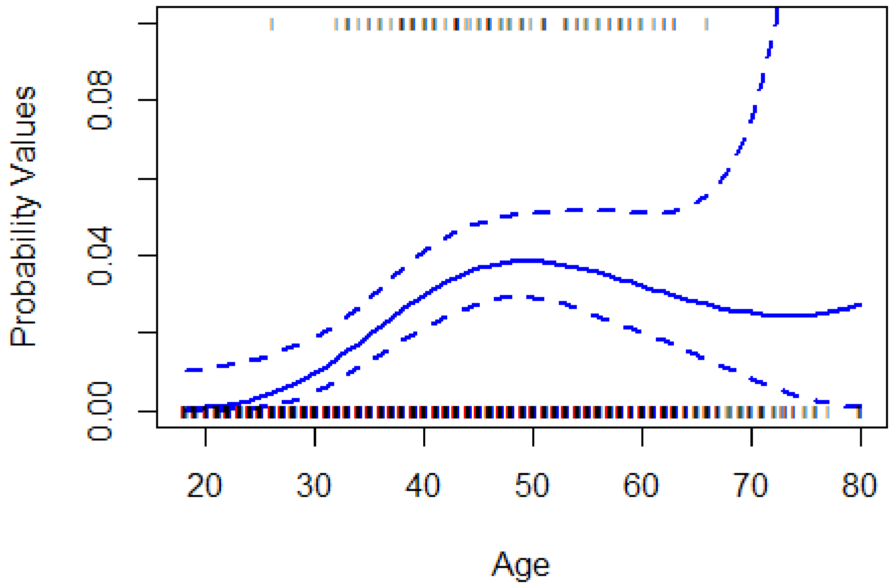
We have done computations on the Logit scale , to convert it to probabilities we will use LateX language which is used in tysetting Mathematical formulas-

This is the formula to compute the probabilities

$$p = \frac{e^\eta}{1 + e^\eta}.$$

```
#comuting the 95% confidence interval for the Fitted Probabilities value
prob.bands = exp(se.band)/ (1 + exp(se.band))
matplot(age.grid,prob.bands,col="blue",lwd = c(2,2,2),lty=c(1,2,2),
        type="l",ylim=c(0,.1),xlab="Age",ylab="Probability Values")

#jitter() function to uniformly add random noise to properly see the
densities
points(jitter(age),I(wage > 250)/10 , pch="I",cex=0.5)
```



The ***blue dotted lines*** represent the 95% Confidence Interval of the fitted Probabilities.

The black dots are the actual Probability values for Binary Response Wage, i.e if wage > 250 is true then 1(TRUE) ,otherwise 0(FALSE).