Stats 506 (F20) Group Project

Group 6: Erin Cikanek, Suppapat Korsurat, Kyle William Schulz

November 13, 2020

Contents

GROUP TO DO LIST

1. Match up language within code between group members how-tos
2. Summary/Discussion/Reference sections
3. Match up plotting style as much as possible
4. Organize git repo
5. Update readme
6. Improve readme style/info

Introduction

Linear regression has become widely known as a backbone of modern statistics. Even as more complex, "black box"-style machine learning techniques increase in popularity, many statisticians and researchers still fall back on regression for its interpretability and simpleness. However, linear regression relies on a number on assumptions that may not always be true in practice, such as the constant, monotonic linearity of predictor variables in relation to the response. In this guide, we explore the use of splines to help model predictor variables that may have changing relationships across their domain. These techniques help us to match the predictive power seen in some more advanced machine learning algorithms while keeping the benefits gained by using regression. We show examples in three popular statistical modelling languages - python, R, and STATA.

Data

In this guide, we will be using the "wage" dataset from the R package ISLR. This data is also used in the book Introduction to Statistical Learning. This dataset contains wages from 3,000 Mid-Atlantic, male workers, between the years 2003-2009, along with a select number of other personal demographics. We retain the variables for `wage`, `age`, `year`, and `education` for our analysis. Our goal is to examine the relationship between age, year, and education and workers' yearly wage.

Method

We will first calculate a simple linear regression as a baseline. We will then implement four different spline-like techniques on the "age" predictor variable: a step function, polynomial regression, basis spline, and natural spline. At each step, we will check for fit quality, noting any potential improvements along the way. We will conclude with a retrospective and summary of what we learned.

Core Analysis

Python

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


#Packages required
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
#%matplotlib inline
import statsmodels.api as sm
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from patsy import dmatrix


# In[2]:


#Let's read in the data file
data = pd.read_csv("/Users/kwschulz/STATS506/Stats506_Project/Dataset/data.csv")


# In[3]:


#Take a quick glance at what the data looks like
data.head()


# In[4]:


#Let's check to see if we have any missing values
data.isna().sum()


# In[5]:


#Filter to variables for our analysis
data = data[["wage", "age", "education", "year"]]


# In[6]:


#Map education to ordinal scale
education_map = {"1. < HS Grad":1,"2. HS Grad":2,
```

```
                "3. Some College":3, "4. College Grad":4,
                "5. Advanced Degree":5}
data['education'] = data.education.map(education_map)


# In[7]:


#Lets check the distribution of our predictors
data[["wage", "age"]].hist(layout=(2,1), figsize=(15,15))
plt.show()
plt.savefig('hist.png')


# In[8]:


#checking year distribution
data.year.value_counts().reindex([2003, 2004, 2005, 2006, 2007, 2008, 2009]).plot(kind='bar',
                                                                                   title='year',
                                                                                   ylabel='count',
                                                                                   figsize=(7.5,7.5))
plt.savefig('year_bar.png')


# In[9]:


#checking education distribution
data.education.value_counts().reindex([1, 2, 3, 4, 5]).plot(kind='bar',
                                                             title='education',
                                                             ylabel='count',
                                                             figsize=(7.5,7.5))
plt.savefig('education_bar.png')


# In[10]:


#linear regression model
model = sm.OLS(data["wage"], sm.add_constant(data.drop('wage',axis=1))).fit()


# In[11]:


#let's check how it did
model.summary()


# In[12]:
```

```
#let's cut age into 6 bins - stepwise
data["age_cut"] = pd.cut(data.age, bins=6, labels=False)


# In[13]:


#now let's model age with bins
model2 = sm.OLS(data["wage"], sm.add_constant(data.drop(['wage','age'],axis=1))).fit()


# In[14]:


#model 2 summary
model2.summary()


# In[15]:


#let's check out the scatter plot of age v wage
data.plot(x="age", y="wage", kind='scatter', figsize=(7.5,7.5))


# In[16]:


#2nd degree polynomial
p = np.poly1d(np.polyfit(data["age"], data["wage"], 2))
t = np.linspace(0, 80, 200)
plt.plot(data["age"], data["wage"], 'o', t, p(t), '-')
rs = sm.OLS(data["wage"],
            np.column_stack([data["age"]**i for i in range(2)]) ).fit().rsquared
plt.title('r2 = {}'.format(rs))
plt.show()
plt.savefig('poly2.png')


# In[17]:


#3rd degree polynomial
p = np.poly1d(np.polyfit(data["age"], data["wage"], 3))
t = np.linspace(0, 80, 200)
plt.plot(data["age"], data["wage"], 'o', t, p(t), '-')
rs = sm.OLS(data["wage"],
            np.column_stack([data["age"]**i for i in range(3)]) ).fit().rsquared
plt.title('r2 = {}'.format(rs))
plt.show()
plt.savefig('poly3.png')
```

```
# In[18]:


#4th degree polynomial
p = np.poly1d(np.polyfit(data["age"], data["wage"], 4))
t = np.linspace(0, 80, 200)
plt.plot(data["age"], data["wage"], 'o', t, p(t), '-')
rs = sm.OLS(data["wage"],
            np.column_stack([data["age"]**i for i in range(4)]) ).fit().rsquared
plt.title('r2 = {}'.format(rs))
plt.show()
plt.savefig('poly4.png')


# In[19]:


#5th degree polynomial
p = np.poly1d(np.polyfit(data["age"], data["wage"], 5))
t = np.linspace(0, 80, 200)
plt.plot(data["age"], data["wage"], 'o', t, p(t), '-')
rs = sm.OLS(data["wage"],
            np.column_stack([data["age"]**i for i in range(5)]) ).fit().rsquared
plt.title('r2 = {}'.format(rs))
plt.show()
plt.savefig('poly5.png')


# In[20]:


#let's do a third polynomial regression
polynomial_features= PolynomialFeatures(degree=3)
age_p = polynomial_features.fit_transform(data['age'].to_numpy().reshape(-1, 1))
model3 = sm.OLS(data["wage"], sm.add_constant(np.concatenate([data[['education', 'year']].to_numpy(), ag


# In[21]:


#check our results
model3.summary(xname=['education', 'year', 'const', 'poly(age, 3)1', 'poly(age, 3)2', 'poly(age, 3)3'])


# In[22]:


#implementing a bspline for age
age_bs = dmatrix("bs(data.age, df=6)",{"data.age": data.age}, return_type='dataframe')
model4 = sm.OLS(data["wage"], pd.concat([age_bs, data[['education', 'year']]], axis=1)).fit()
model4.summary()
```

```
reg wage age year edu

      Source |       SS           df       MS      Number of obs   =     3,000
-------------+----------------------------------   F(3, 2996)      =    342.74
       Model |  1334297.35         3  444765.784   Prob > F        =    0.0000
    Residual |  3887788.36     2,996  1297.65966   R-squared       =    0.2555
-------------+----------------------------------   Adj R-squared   =    0.2548
       Total |  5222085.71     2,999  1741.27566   Root MSE        =    36.023

------------------------------------------------------------------------------
        wage |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   .5812897   .0571732    10.17   0.000      .469187    .6933925
        year |   1.087844   .3249123     3.35   0.001     .4507704    1.724918
        educ |   15.91568   .5425182    29.34   0.000     14.85193    16.97943
       _cons |  -2142.846   651.6029    -3.29   0.001     -3420.48   -865.2114
------------------------------------------------------------------------------
```

Figure 1: OLS output for wage ~ age + year + education

```
# In[23]:


#implementing a natural spline for age
age_ns = dmatrix("cr(data.age, df=6)",{"data.age": data.age}, return_type='dataframe')
model5 = sm.OLS(data["wage"], pd.concat([age_ns, data[['education', 'year']]], axis=1)).fit()
model5.summary()
```

```
"When you want to show only code, but prevent this chunck to run."
```

```
## [1] "When you want this chunck to run, but don't want to show the code."
```

Stata

Before starting analysis using splines, first look at OLS regression with wage as it relates to age, year, and education. We can run the simple code below to look at this relationship.

```
reg wage age year edu
```

Stata will return the following output:

To see if a non-linear relationship might be present, kernal density, pnorm, and qnorm plots can assit with this:

```
predict r, resid
kdensity r, normal
```

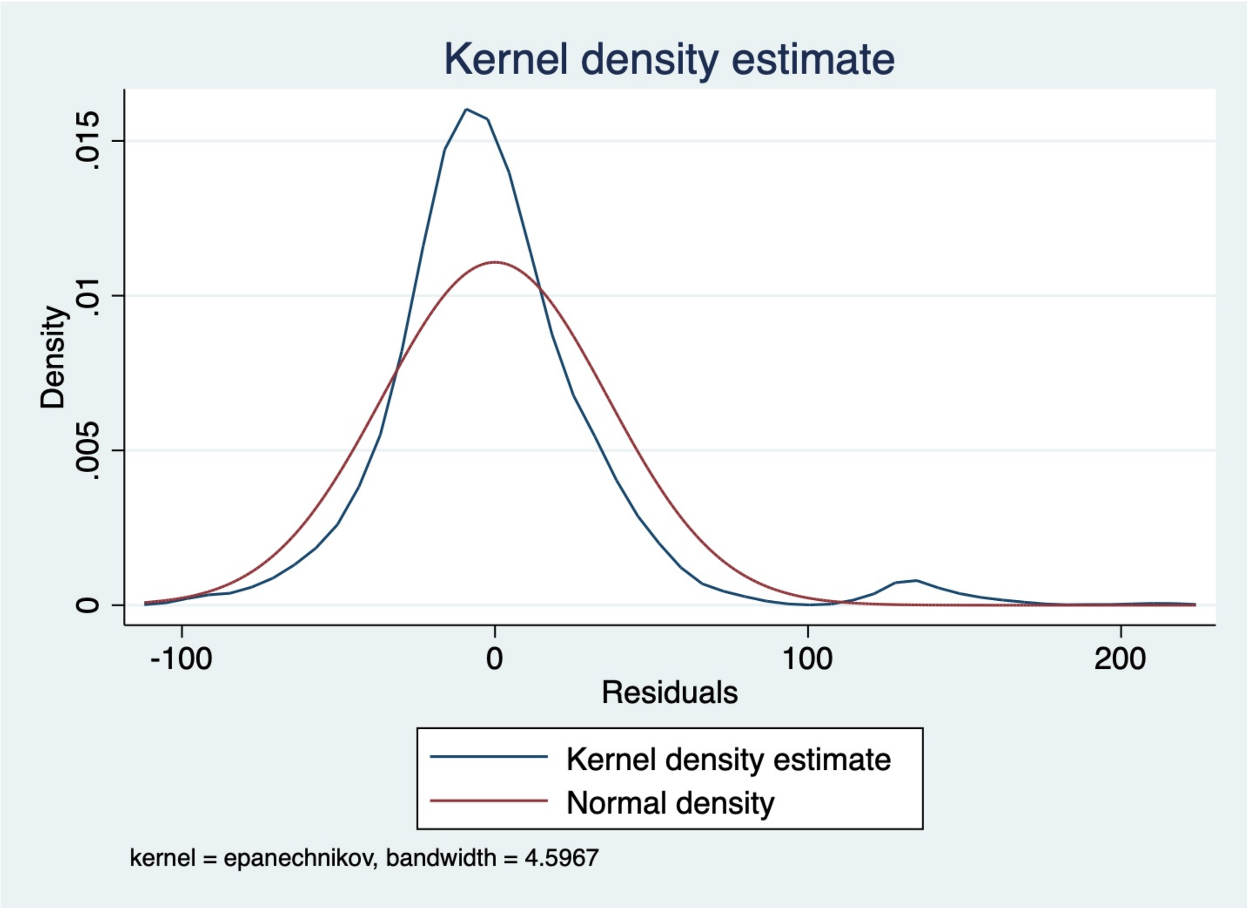
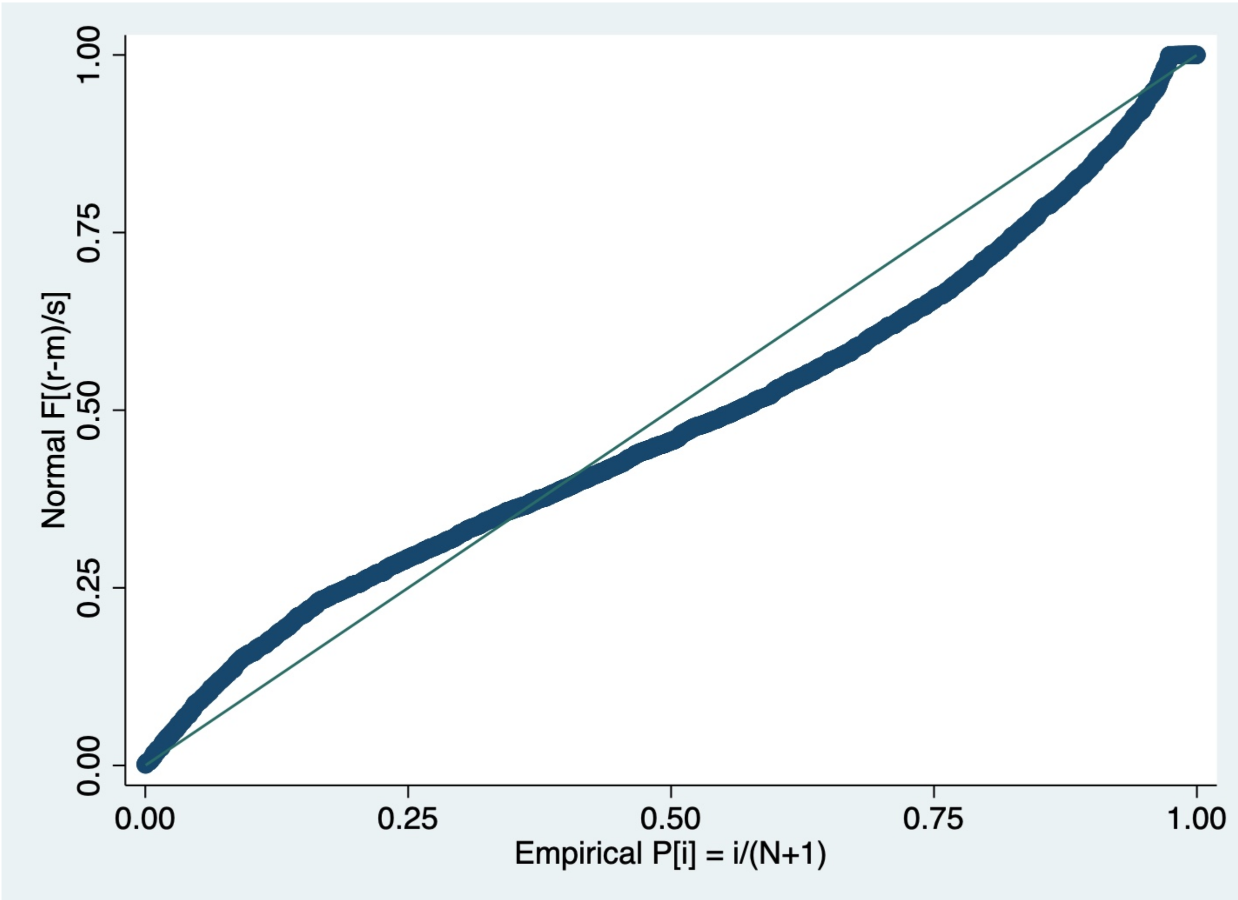
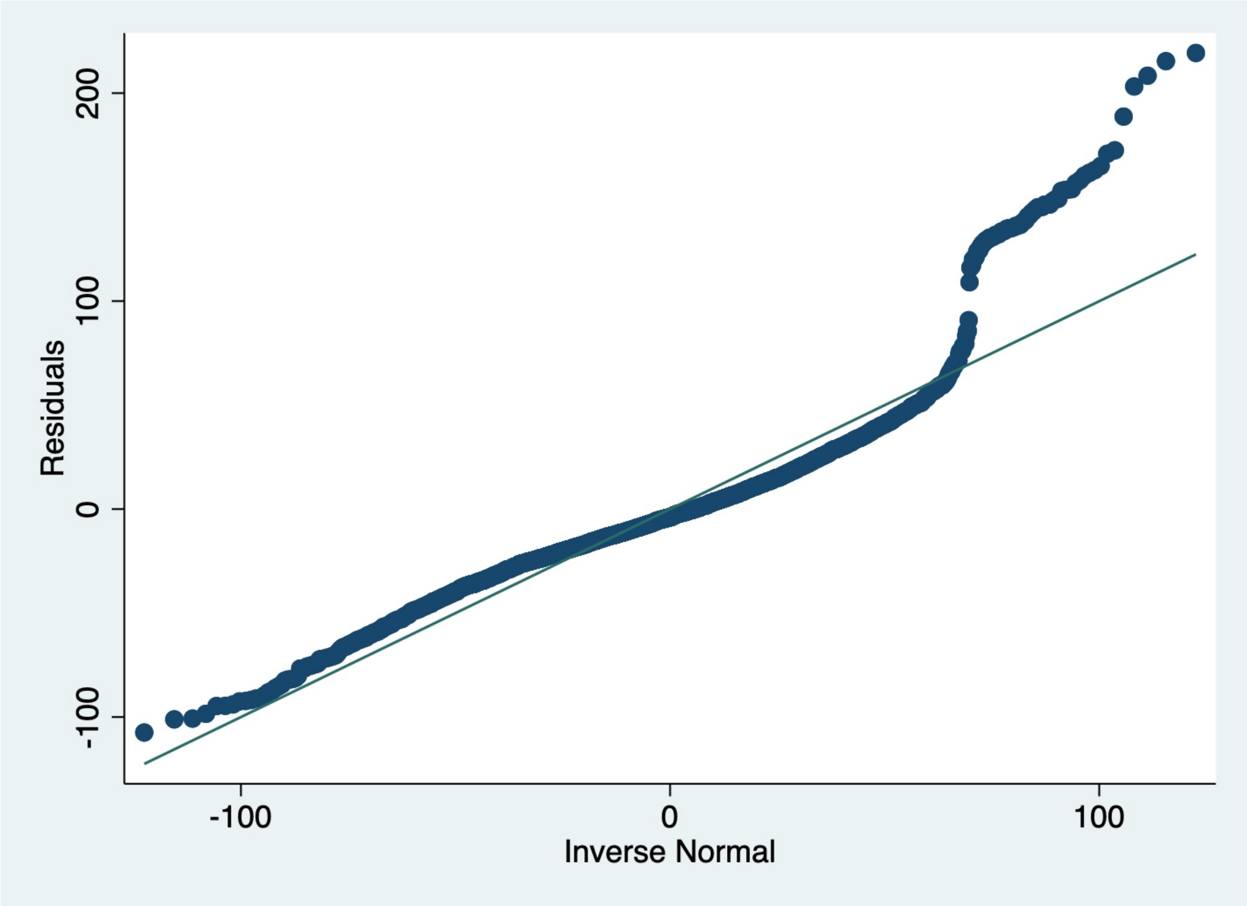


Figure 2: Kernal Density Plot


1.00
0.75
0.50
0.25
0.00
Normal F[(r-m)/s]
0.00
0.25
0.50
0.75
1.00
Empirical P[i] = i/(N+1)



After looking at these plots we might consider the different relationships that age may have with wage. We can plot the two-way fit between wage and age, our main variables of interest, to compare a basic linear, polynomial, and quadratic fit.

```
twoway (scatter wage age) (lfit wage age) (fpfit wage age) (qfit wage age)
```

Based on these plots we might be interested in trying to fit a cubic polynomial plot next.

Cubic Polynomial

To create a cubic polynomial in stata we can use the `##` command with the `age` variable. The regression is written as before with the addition of a cubic fit:

```
reg wage c.age##c.age##c.age year educ
```

The output in Stat will look like this:

Piecewise Step Function Regression

For the piecewise step function, the steps and intercepts in Stata must be determined manually. Based on analysis in `R` we determined that including 6 groups with 5 cutpoints is best. The below code shows how to generate six age categories and their intercepts.

```
* generate 6 age variables, one for each bin *
* the age varaible does not have decimels *

generate age1 = (age - 28.33)
```

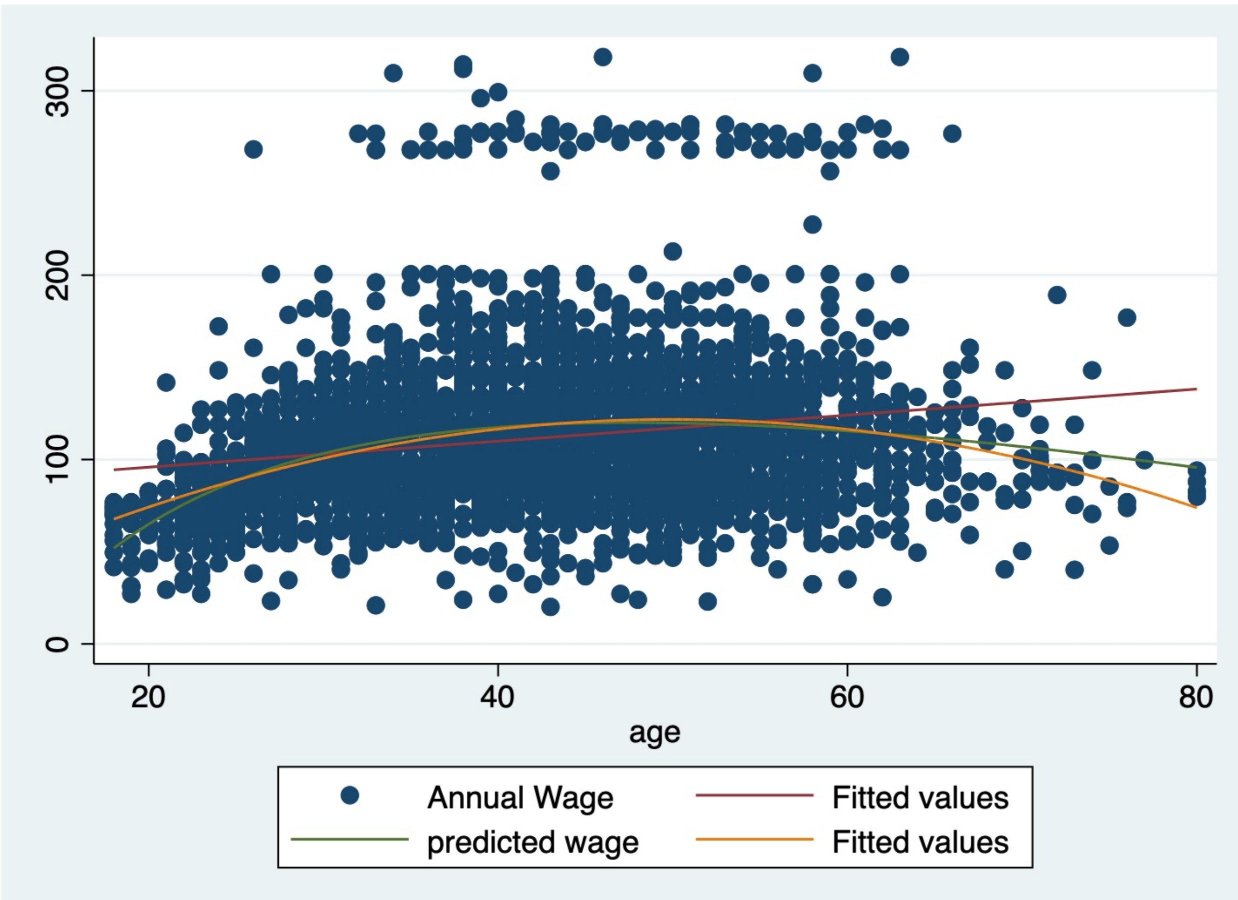


Figure 3: Fitted Plot - Linear (red), polynomial (green), and quadratic (yellow) fit

Source	SS	df	MS
Model	1486051.75	5	297210.35
Residual	3736033.96	2,994	1247.84033
Total	5222085.71	2,999	1741.27566

Number of obs	=	3,000
F(5, 2994)	=	238.18
Prob > F	=	0.0000
R-squared	=	0.2846
Adj R-squared	=	0.2834
Root MSE	=	35.325

wage	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
age	7.405755	1.424342	5.20	0.000	4.612967	10.19854
c.age#c.age	-.1163401	.0326773	-3.56	0.000	-.1804124	-.0522678
c.age#c.age#c.age	.0005453	.0002394	2.28	0.023	.0000759	.0010148
year	1.194394	.318829	3.75	0.000	.5692475	1.819539
educ	15.29865	.5351343	28.59	0.000	14.24938	16.34792
_cons	-2470.347	640.3507	-3.86	0.000	-3725.919	-1214.775

Figure 4: Regression with Cubic polynomial for Age

```
replace age1 = 0 if (age >= 28.33)
generate age2 = (age-38.66)
replace age2 = 0 if age <28.33 | age > 38.66
generate age3 = (age- 48.99)
replace age3 = 0 if age <38.66 | age >=48.99
generate age4 = (age - 59.33)
replace age4 = 0 if age <48.99 | age >= 59.33
generate age5 = (age - 69.66)
replace age5= 0 if age < 59.33 | age>=69.66
generate age6 = (age-80)
replace age6 = 0 if age <69.66

* create intercept variables*


generate int1 = 1
replace int1 = 0 if age >= 28.33
generate int2 = 1
replace int2 = 0 if age <28.33 | age > 38.66
generate int3 = 1
replace int3 = 0 if age <38.66 | age >=48.99
generate int4 = 1
replace int4 = 0 if age <48.99 | age >= 59.33
generate int5 = 1
replace int5= 0 if age < 59.33 | age>=69.66
generate int6 = 1
replace int6 = 0 if age <69.66
```

Using these variables we can then compute a step-wise regression.

Source	SS	df	MS			
				Number of obs	=	3,000
				F(13, 2986)	=	92.98
Model	1504777.18	13	115752.09	Prob > F	=	0.0000
Residual	3717308.53	2,986	1244.91244	R-squared	=	0.2882
				Adj R-squared	=	0.2851
Total	5222085.71	2,999	1741.27566	Root MSE	=	35.283

wage	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
int1	-2472.164	641.5494	-3.85	0.000	-3730.087	-1214.24
int2	-2450.95	641.4461	-3.82	0.000	-3708.671	-1193.229
int3	-2455.446	641.5612	-3.83	0.000	-3713.392	-1197.499
int4	-2456.055	641.5833	-3.83	0.000	-3714.045	-1198.064
int5	-2465.509	641.8945	-3.84	0.000	-3724.11	-1206.909
int6	-2473.924	641.2264	-3.86	0.000	-3731.214	-1216.633
age1	2.705993	.6468721	4.18	0.000	1.437632	3.974353
age2	2.680931	.4603949	5.82	0.000	1.778208	3.583654
age3	-.0539289	.4000442	-0.13	0.893	-.838319	.7304612
age4	.1505525	.4248549	0.35	0.723	-.6824854	.9835904
age5	-1.15859	1.104284	-1.05	0.294	-3.323824	1.006644
age6	.0387909	1.927735	0.02	0.984	-3.741033	3.818615
year	1.260048	.3198308	3.94	0.000	.6329367	1.887159
educ	15.31426	.5367711	28.53	0.000	14.26178	16.36674

Figure 5: Step-wise regression for Age with 6 bins

```
regress wage int1 int2 int3 int4 int5 int6 age1 age2 age3 age4 age5 age6 ///
    year educ, hascons
```

After running the regression we can then use the predicted yhats to graph the results:

```
predict yhat

twoway (scatter wage age, sort) ///
         (line yhat age if age <28.33, sort) ///
         (line yhat age if age >=28.33 & age < 38.66, sort) ///
         (line yhat age if age >=38.66 & age < 48.99, sort) ///
         (line yhat age if age >=48.99 & age<59.33, sort) ///
         (line yhat age if age >=59.33 & age<69.66, sort) ///
         (line yhat age if age >=69.66, sort), xline(28.33 38.66 48.99 59.33 69.66) // this looks awful
```

Basis Spline

For the basis spline, we use the command `bspline`, created by Roger Newson and suggested by [Germán Rodríguez at Princeton] (https://data.princeton.edu/eco572/smoothing2). To create the spline, we call `bspline`, setting the x variable to age and then identifying where we would like the knots in the function. For this example I use 3 knots at 35, 50 and 65, however it should be noted that the min and max of the values need to be included in the knots parentheses. I also use a cubic spline, incidated by `p(3)`. The last step in

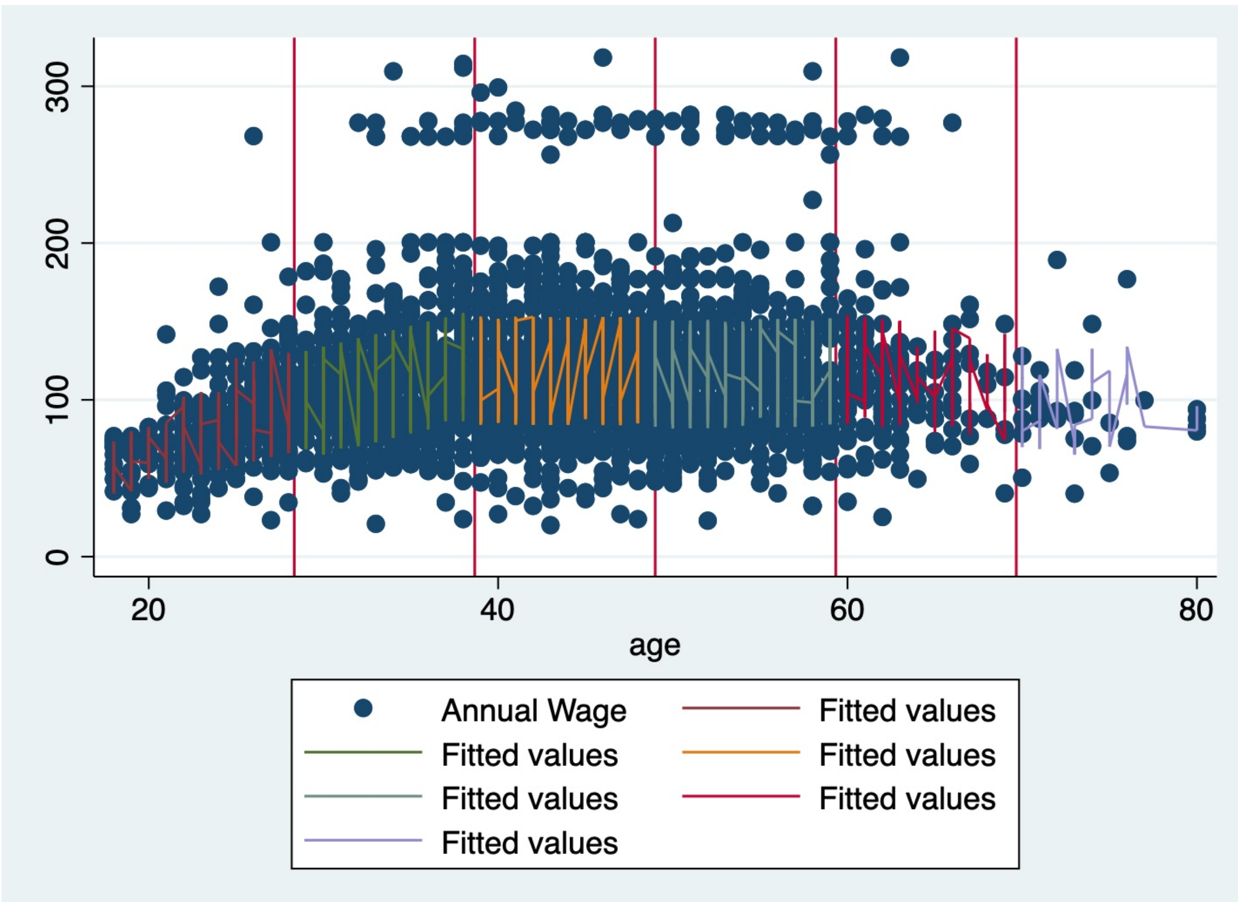


Figure 6: Step-wise regression for Age with 6 bins

Source	SS	df	MS			
				Number of obs	=	3,000
				F(9, 2991)	=	3472.27
Model	38929232.8	9	4325470.32	Prob > F	=	0.0000
Residual	3725941.05	2,991	1245.7175	R-squared	=	0.9126
				Adj R-squared	=	0.9124
Total	42655173.9	3,000	14218.3913	Root MSE	=	35.295

wage	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
_agespt1	-2408.356	640.3538	-3.76	0.000	-3663.934	-1152.777
_agespt2	-2482.988	641.1796	-3.87	0.000	-3740.186	-1225.79
_agespt3	-2412.551	640.3912	-3.77	0.000	-3668.202	-1156.899
_agespt4	-2424.917	640.7662	-3.78	0.000	-3681.304	-1168.53
_agespt5	-2415.806	640.5638	-3.77	0.000	-3671.796	-1159.815
_agespt6	-2463.628	641.744	-3.84	0.000	-3721.933	-1205.324
_agespt7	-2363.85	649.0144	-3.64	0.000	-3636.409	-1091.29
year	1.242581	.3193946	3.89	0.000	.6163254	1.868836
educ	15.31363	.536419	28.55	0.000	14.26184	16.36542

Figure 7: Basis Spline Regression

the line of code is the code that generates the splines for inclusions in the regression. Then the regression can be written as below.

```
bspline, xvar(age) knots(18 35 50 65 80) p(3) gen(_agespt)


regress wage _agespt* year educ, noconstant
```

The output for the regression in Stata is:

To look at the fit for age, we can examine the two-way scatter plot between `wage` and `age` using the predicted values of the bivariate regression with splines.

```
regress wage _agespt*, noconstant
predict agespt
*(option xb assumed; fitted values)


twoway (scatter wage age)(line agespt age, sort), legend(off)  ///
          title(Basis Spline for Age)
```

Natural Spline

This further extension is still being coded. Please see the README.md file.

R

The library `splines` is required for implementing splines by using R.

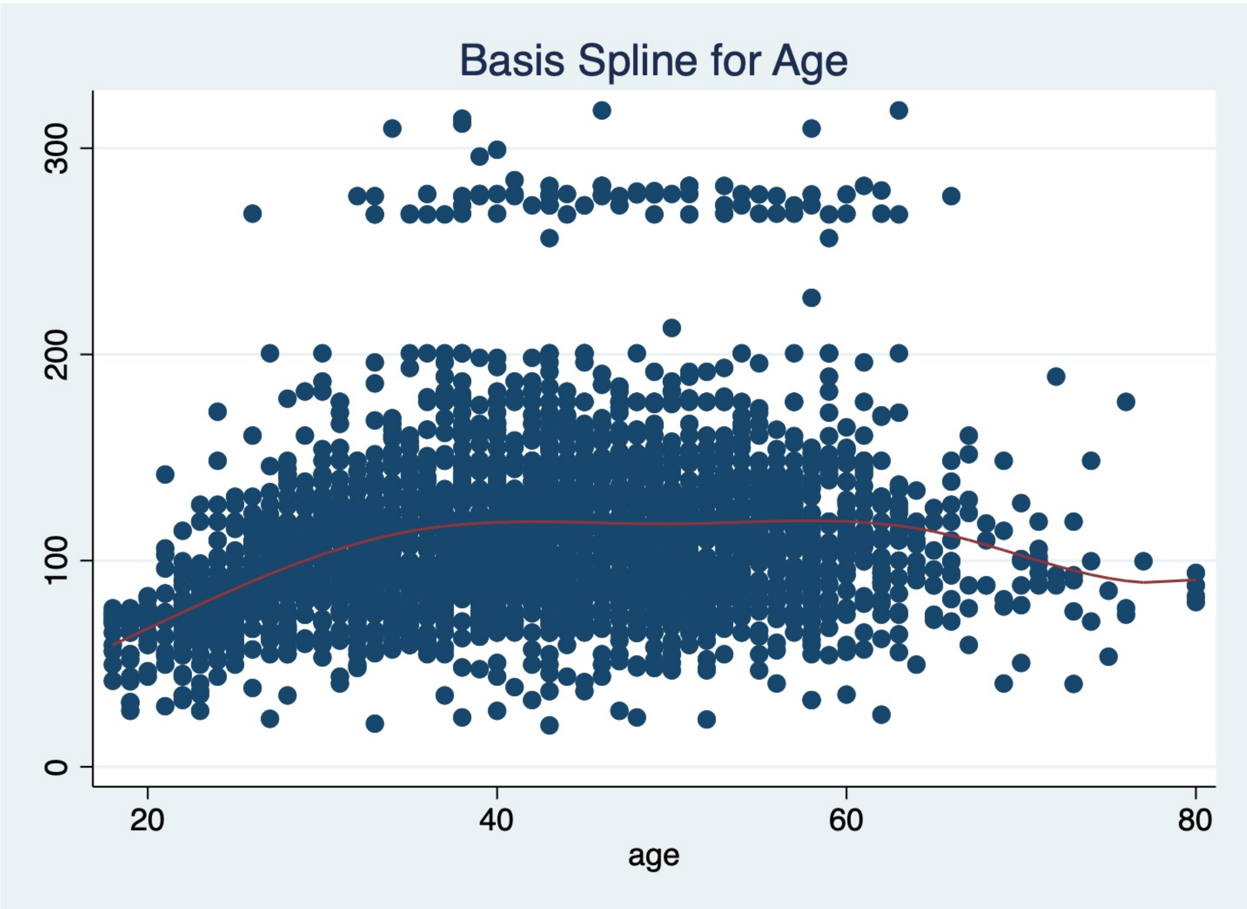


Figure 8: Step-wise regression for Age with 6 bins

```r
library(splines)
```

First, considering the linear regression.

```r
model <- lm(wage ~ age + education + year, data = data)
summary(model)
```

```
##
## Call:
## lm(formula = wage ~ age + education + year, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -113.323  -19.521   -3.964   14.438  219.172
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -2.058e+03  6.493e+02  -3.169  0.00154 **
## age                         5.621e-01  5.714e-02   9.838  < 2e-16 ***
## education2. HS Grad         1.140e+01  2.476e+00   4.603 4.34e-06 ***
## education3. Some College    2.423e+01  2.606e+00   9.301  < 2e-16 ***
## education4. College Grad    3.974e+01  2.586e+00  15.367  < 2e-16 ***
## education5. Advanced Degree  6.485e+01  2.804e+00  23.128  < 2e-16 ***
## year                        1.056e+00  3.238e-01   3.262  0.00112 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.89 on 2993 degrees of freedom
## Multiple R-squared:  0.2619, Adjusted R-squared:  0.2604
## F-statistic:   177 on 6 and 2993 DF,  p-value: < 2.2e-16
```

The R^2 is 0.2619, which is pretty low. Consider the scatter plot between `Wage` and `Age`.

The scatter plot show that the relationship between these two variables are not linear. Hence, we will try various types of spline.

Step Function

Consider applying the step function on `Age`.

```r
model_cut <- lm(wage ~ cut(age, 4) + education + year, data = data)
summary(model_cut)
```

```
##
## Call:
## lm(formula = wage ~ cut(age, 4) + education + year, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -120.260  -19.442   -3.744   14.441  214.958
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -2408.5219   641.1663  -3.756 0.000176 ***
## cut(age, 4)(33.5,49]          20.9265     1.6085  13.010  < 2e-16 ***
## cut(age, 4)(49,64.5]          19.3732     1.8197  10.646  < 2e-16 ***
```

Figure 9: Figure 3.1 Scatter plot between Wage and Age

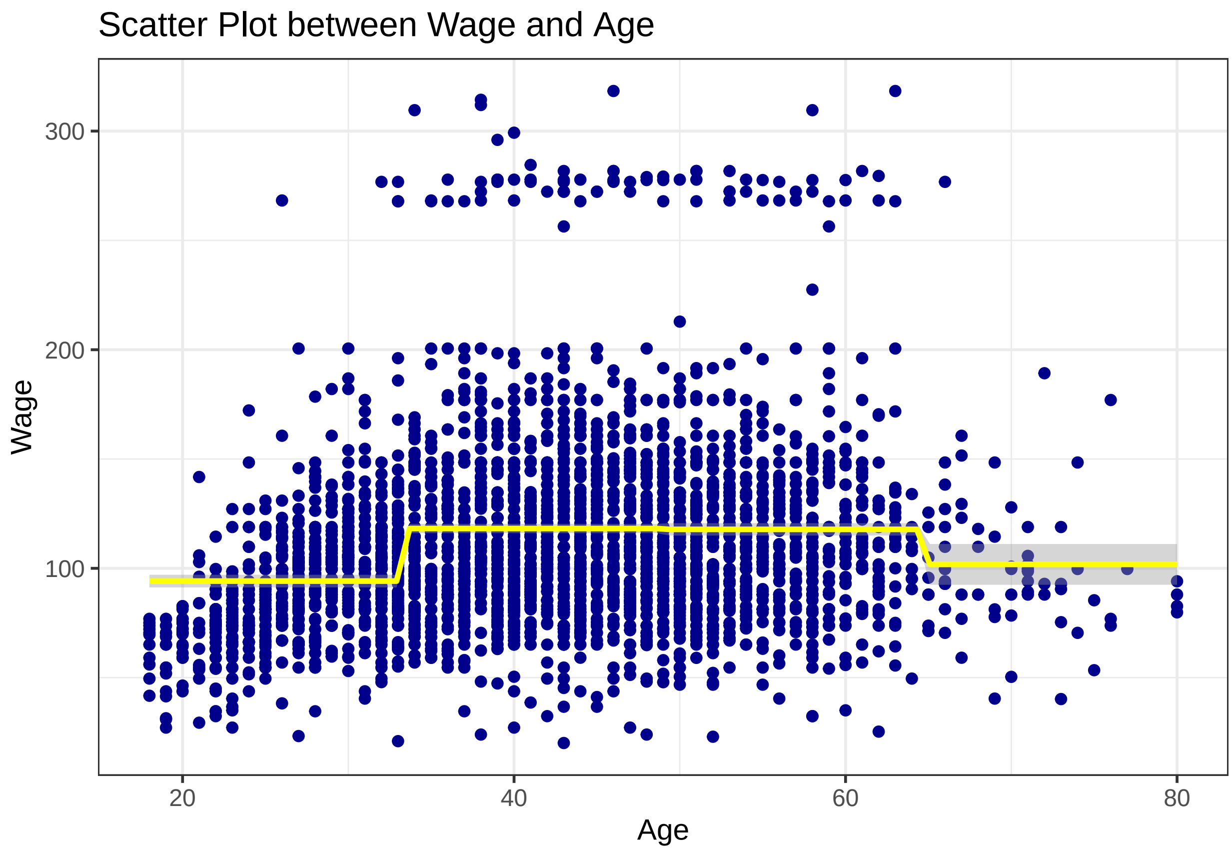


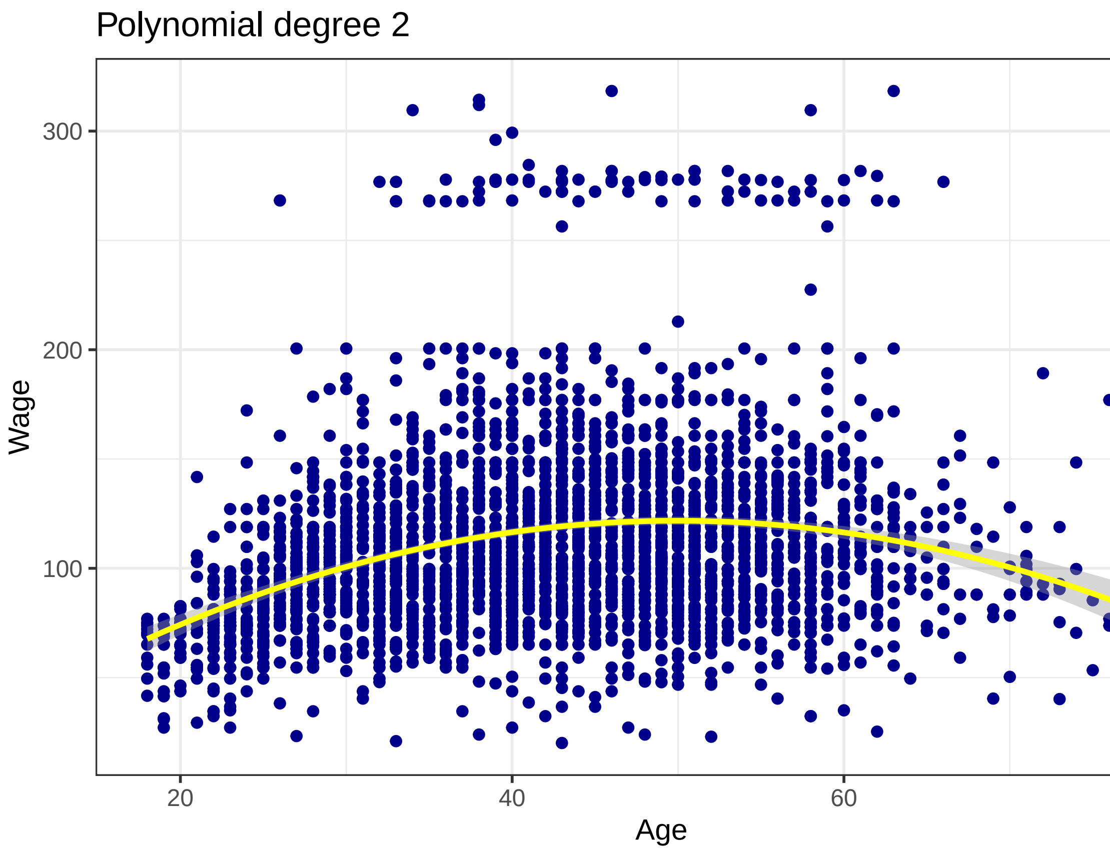
Figure 10: Figure 3.2 Scatter plot between Wage and Age with the step function.

```
## cut(age, 4)(64.5,80.1]          8.0516     4.3783   1.839 0.066014 .
## education2. HS Grad            11.1534     2.4436   4.564 5.21e-06 ***
## education3. Some College       24.1620     2.5739   9.387  < 2e-16 ***
## education4. College Grad       39.2164     2.5533  15.359  < 2e-16 ***
## education5. Advanced Degree    64.1642     2.7675  23.185  < 2e-16 ***
## year                            1.2356     0.3197   3.865 0.000113 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.39 on 2991 degrees of freedom
## Multiple R-squared:  0.2828, Adjusted R-squared:  0.2809
## F-statistic: 147.4 on 8 and 2991 DF,  p-value: < 2.2e-16
```

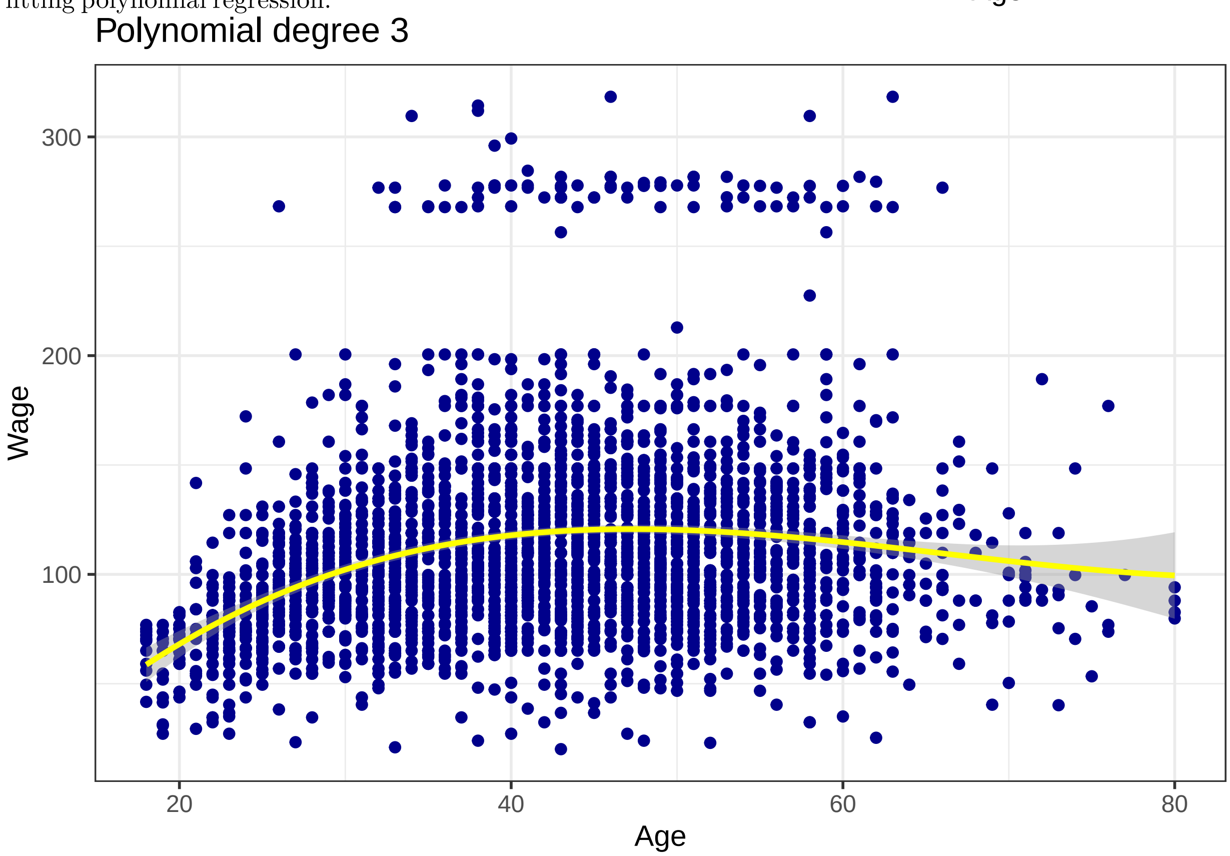
The R^2 is 0.2828, which improved from the previous model. The plot below is a scatterplot between `Wage` and `Age`, also the yellow line represents the step function.

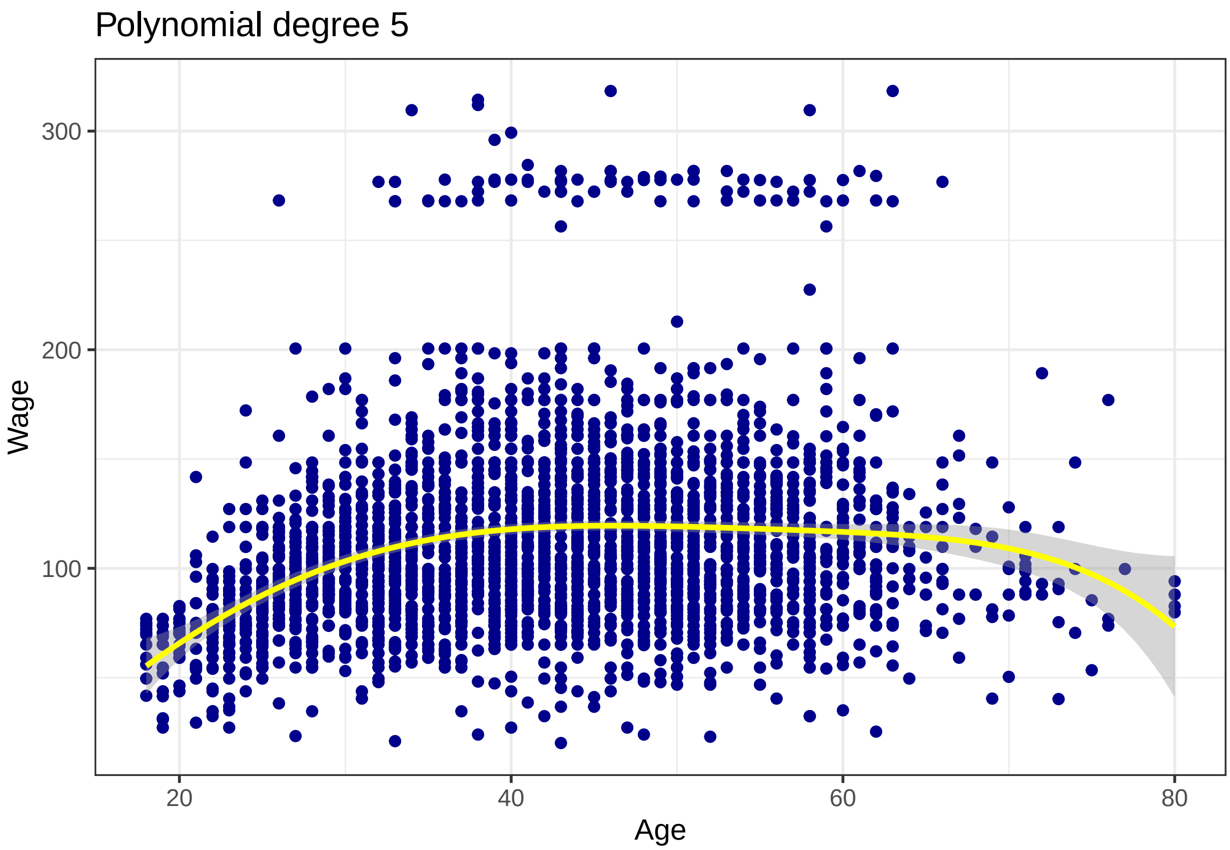
Polynomial Regression

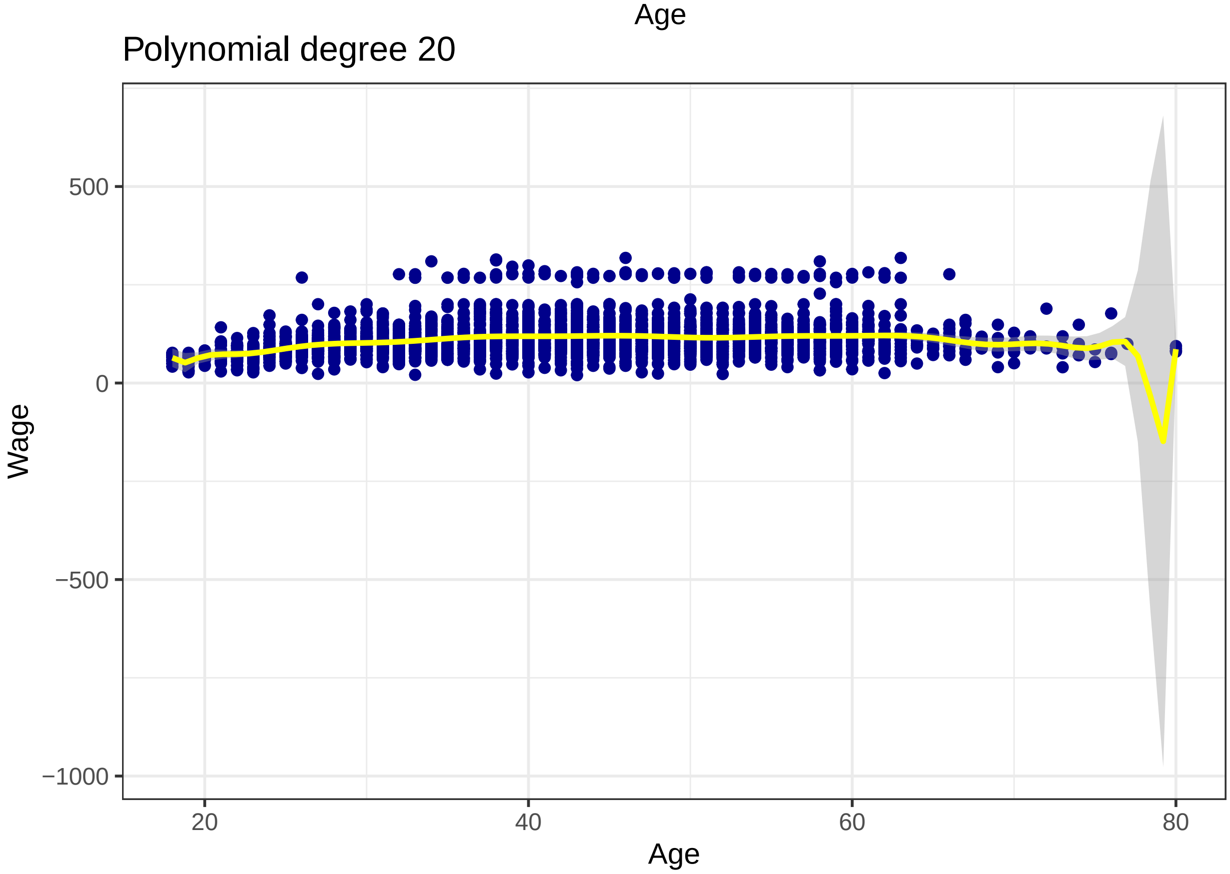
Consider the various number for the degree in the polynomial regression. The plots below are the result from the



fitting polynomial regression.








```
##     Degree of the age polynomial R-Squared
```

```
## 1                             2 0.2896871
## 2                             3 0.2908565
## 3                             4 0.2908565
## 4                             5 0.2914362
## 5                             6 0.2918935
## 6                             7 0.2928255
## 7                             8 0.2928256
## 8                             9 0.2935562
## 9                            10 0.2937707
## 10                           11 0.2937954
## 11                           12 0.2937982
## 12                           13 0.2938966
## 13                           14 0.2940063
## 14                           15 0.2941473
## 15                           16 0.2942057
## 16                           17 0.2947922
## 17                           18 0.2947927
## 18                           19 0.2948218
## 19                           20 0.2948309
```

Even the higher degree give the higher R^2, the `overfitting` problem may be occured. Hence, polynomial regression with degree 3 would be appropriate.

```r
model_poly <- lm(wage ~ poly(age, 3) + education + year, data = data)
summary(model_poly)
```

```
##
## Call:
## lm(formula = wage ~ poly(age, 3) + education + year, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -118.565  -19.789   -3.339   14.399  213.276
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -2247.6445   637.2171  -3.527 0.000426 ***
## poly(age, 3)1                358.1166    35.4147  10.112  < 2e-16 ***
## poly(age, 3)2               -383.1188    35.3679 -10.832  < 2e-16 ***
## poly(age, 3)3                 78.2802    35.2489   2.221 0.026440 *
## education2. HS Grad           10.8127     2.4290   4.452 8.84e-06 ***
## education3. Some College      23.2840     2.5564   9.108  < 2e-16 ***
## education4. College Grad      37.8823     2.5414  14.906  < 2e-16 ***
## education5. Advanced Degree   62.4402     2.7584  22.636  < 2e-16 ***
## year                           1.1633     0.3177   3.662 0.000255 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.19 on 2991 degrees of freedom
## Multiple R-squared:  0.2909, Adjusted R-squared:  0.289
## F-statistic: 153.3 on 8 and 2991 DF,  p-value: < 2.2e-16
```

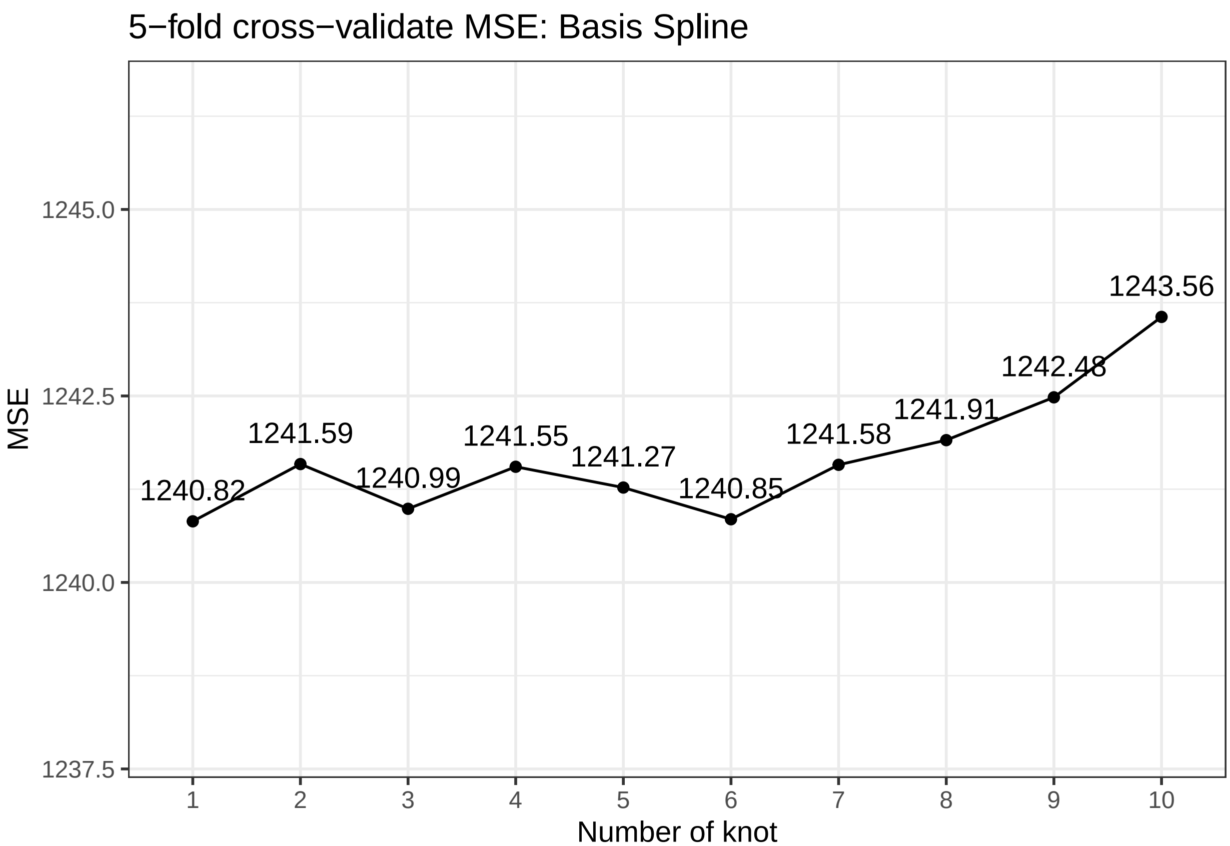
The R^2 is 0.2909, which improved from all previous models.

Basis Spline and Natural Spline

For both `Basis Spline` and `Natural Spline`, the number of knots or the degree of freedom need to be specified. One of the method used for specified is performing `K-fold Cross Validation`. In this case, K is equal to 5. For both types of spline, the highest degree of polynomial for age is 3.

- Basis Spline: df = 4 + knots
- Natural Spline: df = 2 + knots

Consider the MSE for basis spline.



The MSE is lowest when the number of `knot` is equal to 2. Fit the regression with basis spline.

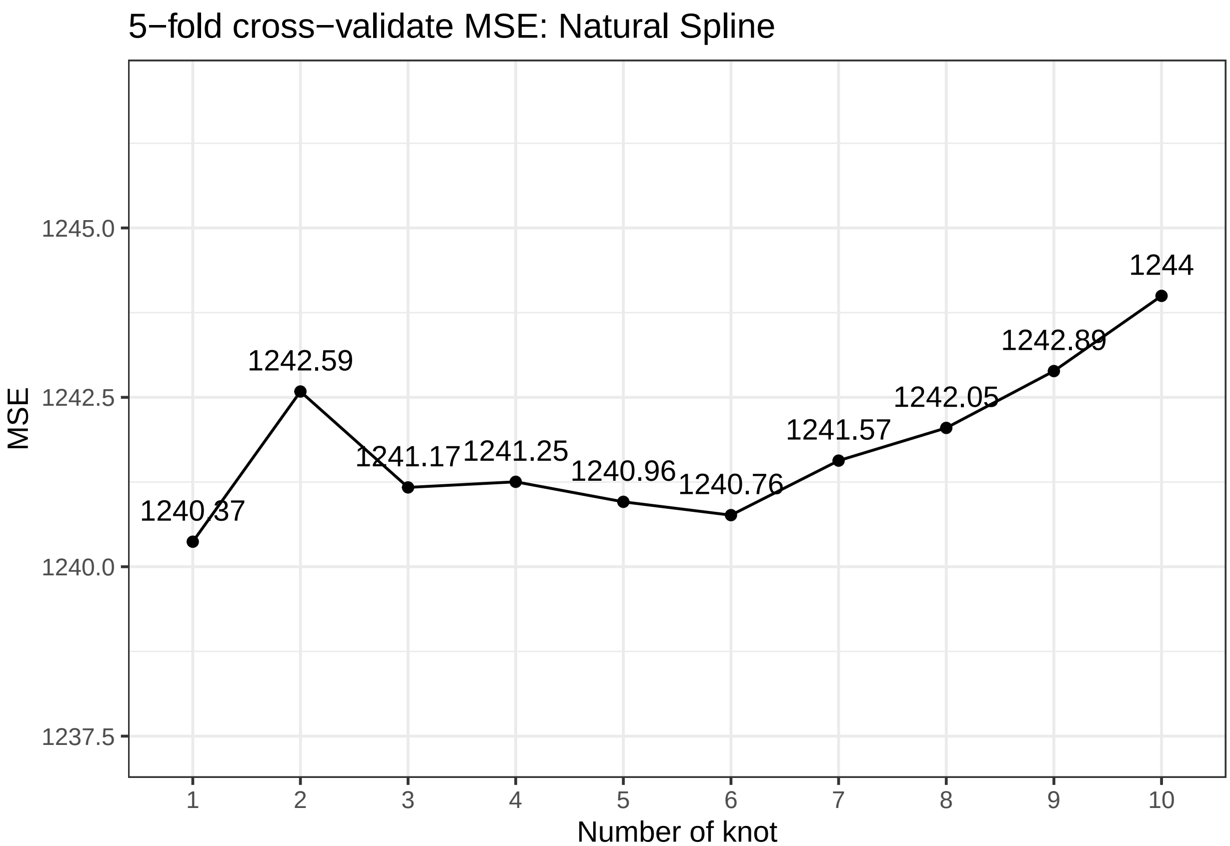
```
model_basis <- lm(wage ~ bs(age, df = 6) + education + year, data = data)
summary(model_basis)
```

```
##
## Call:
## lm(formula = wage ~ bs(age, df = 6) + education + year, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -120.371  -19.640   -3.273   14.086  213.170
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -2344.664    637.830  -3.676 0.000241 ***
## bs(age, df = 6)1                11.675     10.997   1.062 0.288473
## bs(age, df = 6)2                31.678      6.333   5.002 6.01e-07 ***
## bs(age, df = 6)3                46.964      7.371   6.372 2.16e-10 ***
## bs(age, df = 6)4                34.013      7.742   4.393 1.16e-05 ***
```

```
## bs(age, df = 6)5               48.731     12.143   4.013 6.14e-05 ***
## bs(age, df = 6)6                6.633     14.292   0.464 0.642610
## education2. HS Grad            11.075      2.430   4.557 5.41e-06 ***
## education3. Some College       23.638      2.562   9.227  < 2e-16 ***
## education4. College Grad       38.242      2.548  15.008  < 2e-16 ***
## education5. Advanced Degree    62.597      2.761  22.669  < 2e-16 ***
## year                            1.194      0.318   3.753 0.000178 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.17 on 2988 degrees of freedom
## Multiple R-squared:  0.2923, Adjusted R-squared:  0.2897
## F-statistic: 112.2 on 11 and 2988 DF,  p-value: < 2.2e-16
```

The R^2 is 0.2923.

Then consider the Natural Spline.



The MSE is lowest when the number of `knot` is equal to 4. Fit the regression with natural spline.

```
model_natural <- lm(wage ~ ns(age, df = 6) + education + year, data = data)
summary(model_natural)
```

```
##
## Call:
## lm(formula = wage ~ ns(age, df = 6) + education + year, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -121.403  -19.727   -3.143   14.174  214.340
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -2394.4450   638.1274  -3.752 0.000179 ***
## ns(age, df = 6)1                38.7338     4.6496   8.331  < 2e-16 ***
## ns(age, df = 6)2                46.4652     5.8970   7.879 4.57e-15 ***
## ns(age, df = 6)3                38.1178     5.1218   7.442 1.29e-13 ***
## ns(age, df = 6)4                37.0673     4.8062   7.712 1.67e-14 ***
## ns(age, df = 6)5                48.9899    11.6639   4.200 2.75e-05 ***
## ns(age, df = 6)6                 4.3620     8.9214   0.489 0.624922
## education2. HS Grad             11.1264     2.4295   4.580 4.85e-06 ***
## education3. Some College        23.6491     2.5595   9.240  < 2e-16 ***
## education4. College Grad        38.3108     2.5454  15.051  < 2e-16 ***
## education5. Advanced Degree     62.5971     2.7605  22.676  < 2e-16 ***
## year                             1.2186     0.3182   3.830 0.000131 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.16 on 2988 degrees of freedom
## Multiple R-squared:  0.2927, Adjusted R-squared:  0.2901
## F-statistic: 112.4 on 11 and 2988 DF,  p-value: < 2.2e-16
```

The R^2 is 0.2927.

Summary

Discussion

Reference