Beyond Linearity

Vahid Partovi Nia

Lecture 07

McGill

Outline

1. Splines
2. Nearest Neighbours
3. Local Regression

Splines

Nearest Neighbours

Local Regression

Splines

Splines: another way of polynomial regression

univariate function approximation

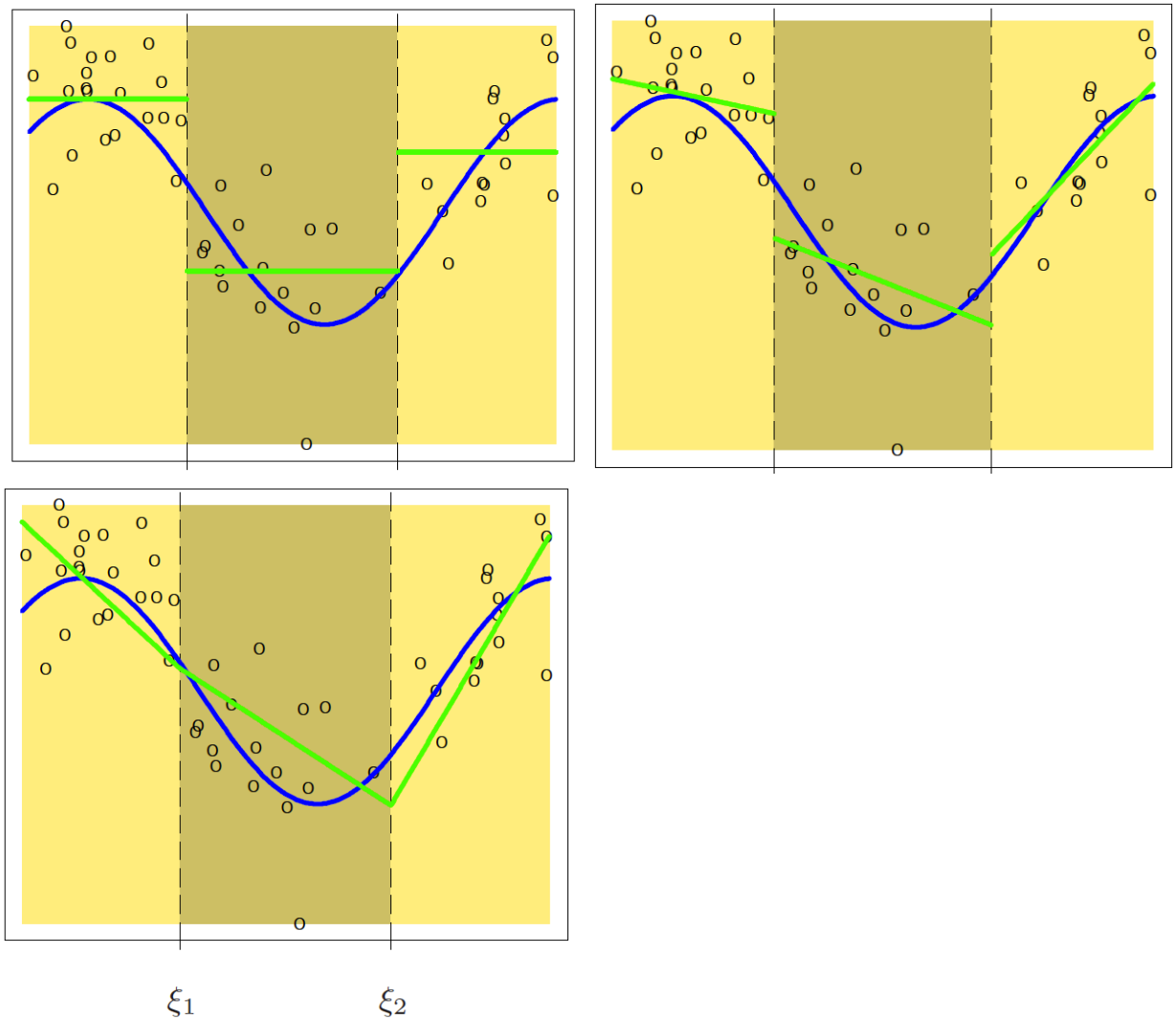
Suppose approximation of a good univariate function over a set of observed $(x_i, y_i), i = 1, \ldots, n$.

$$y_i = f(x_i) + \varepsilon_i \approx \sum_j \beta_j b_j(x_i)$$

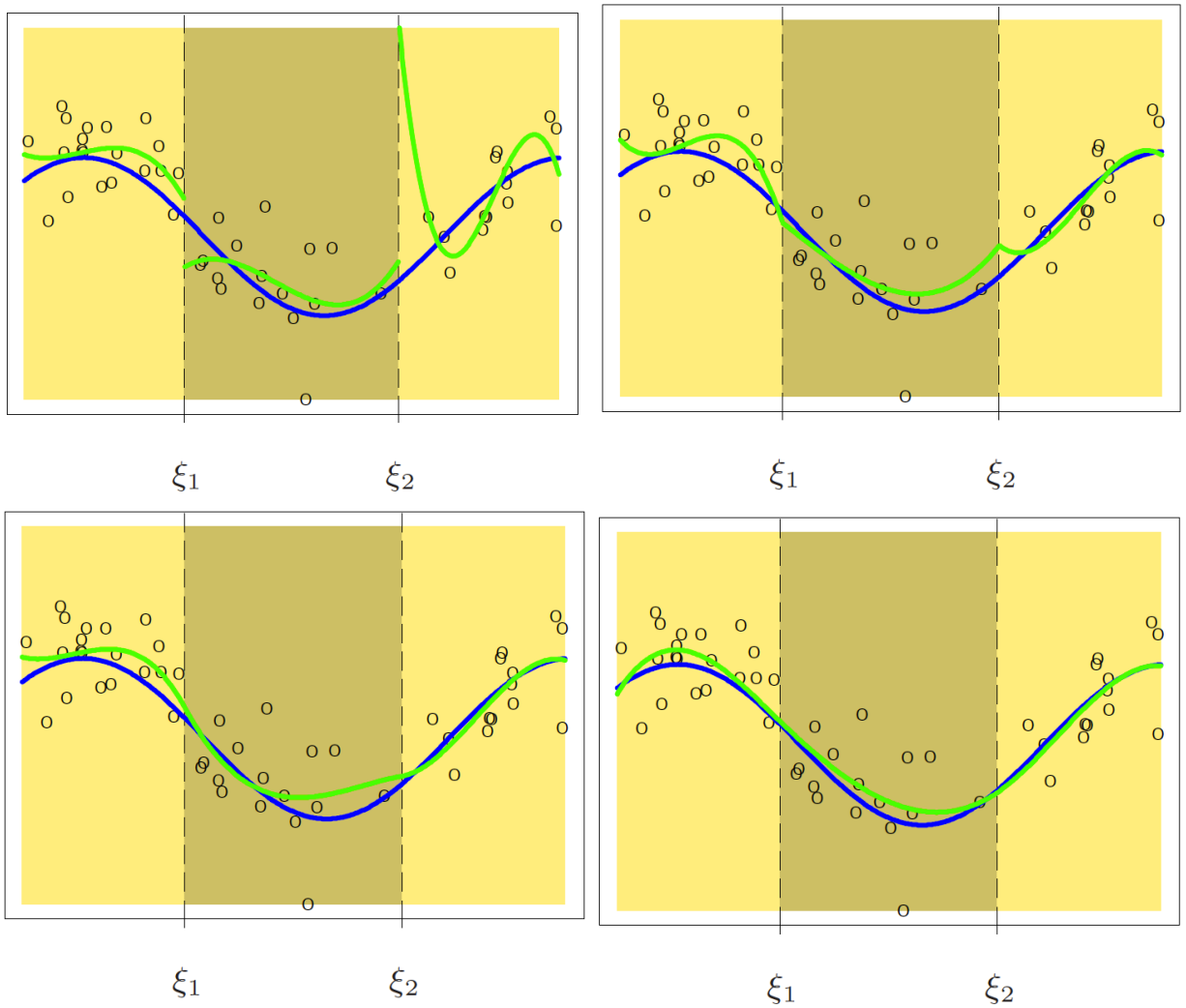
- polynomial base $b_j(x_i) = x_i^j$
- Fourier base
- Wavelet base

$$y_i \approx \sum_{j=1}^{2^k-1} \beta_j^{(k)} b_j^{(k)}(x_i)$$

Piecewise polynomials



Cubic



B splines of order m are piece-wise polynomial functions

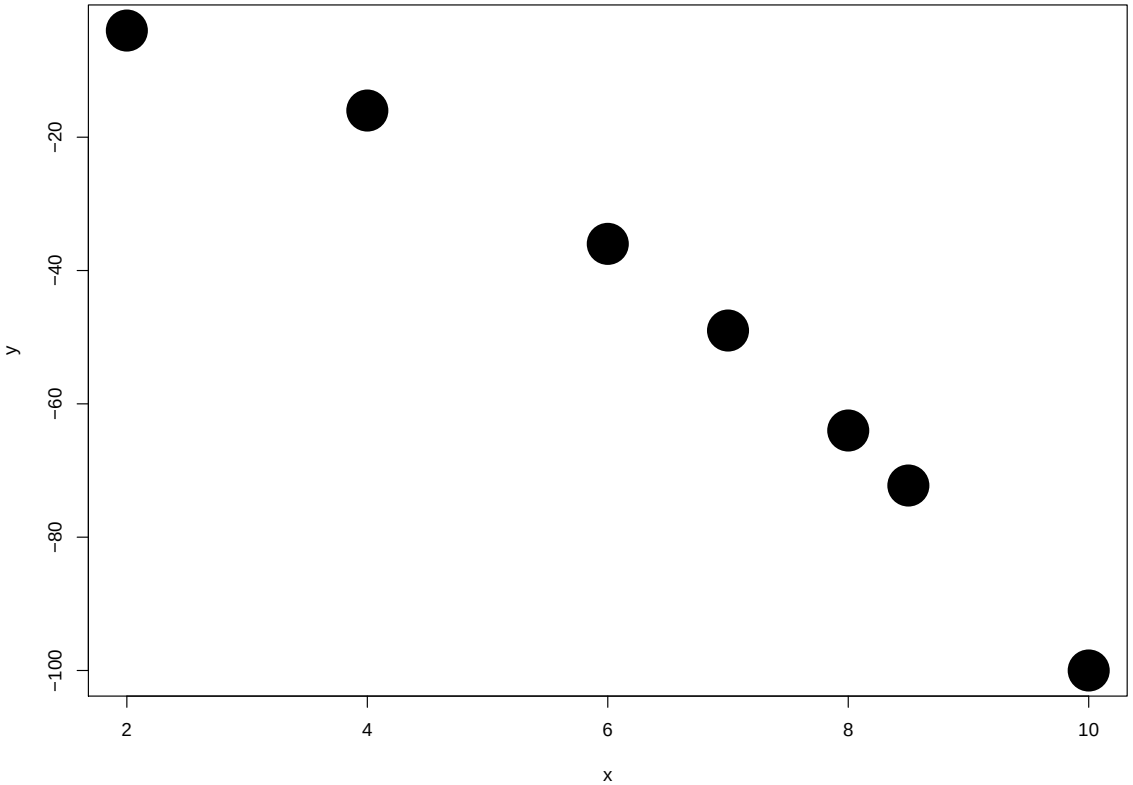
$$
\begin{aligned}
h_0(x) &= 1 \\
h_1(x) &= x \\
h_2(x) &= x^2 \\
h_{2+l}(x) &= \{\max(0, x - \xi_l)\}^3
\end{aligned}
$$

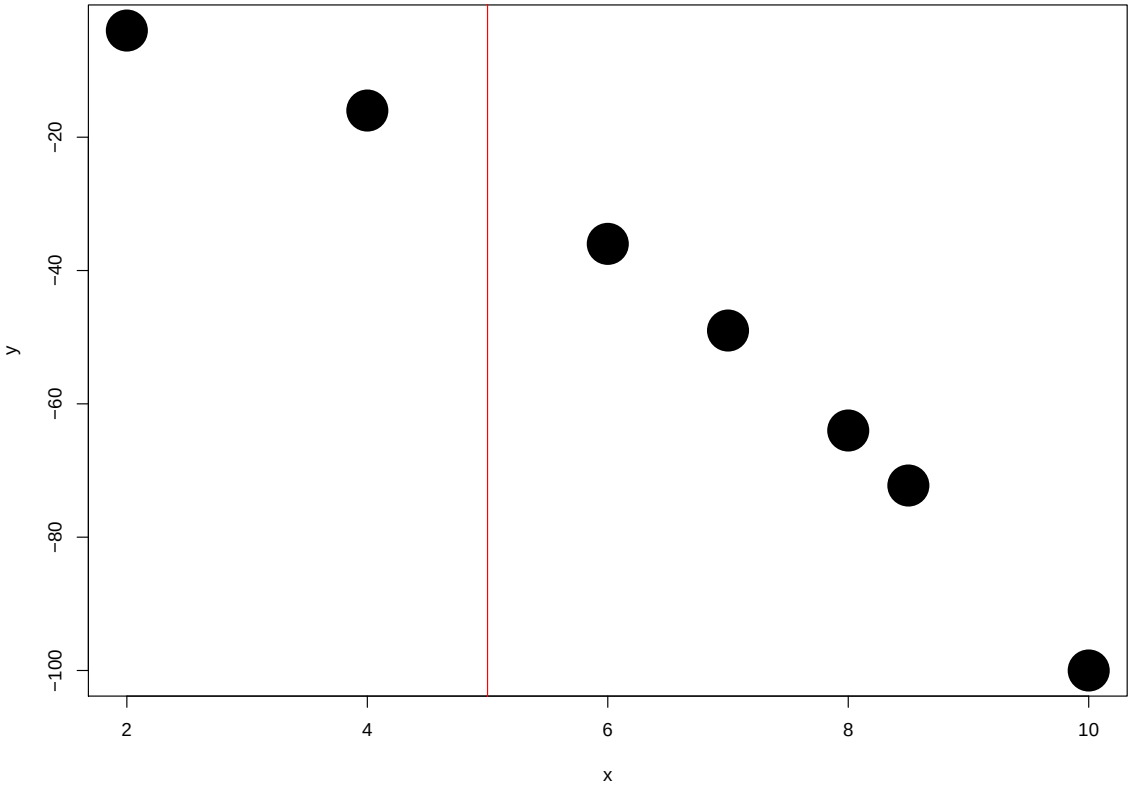
Generalized ridge

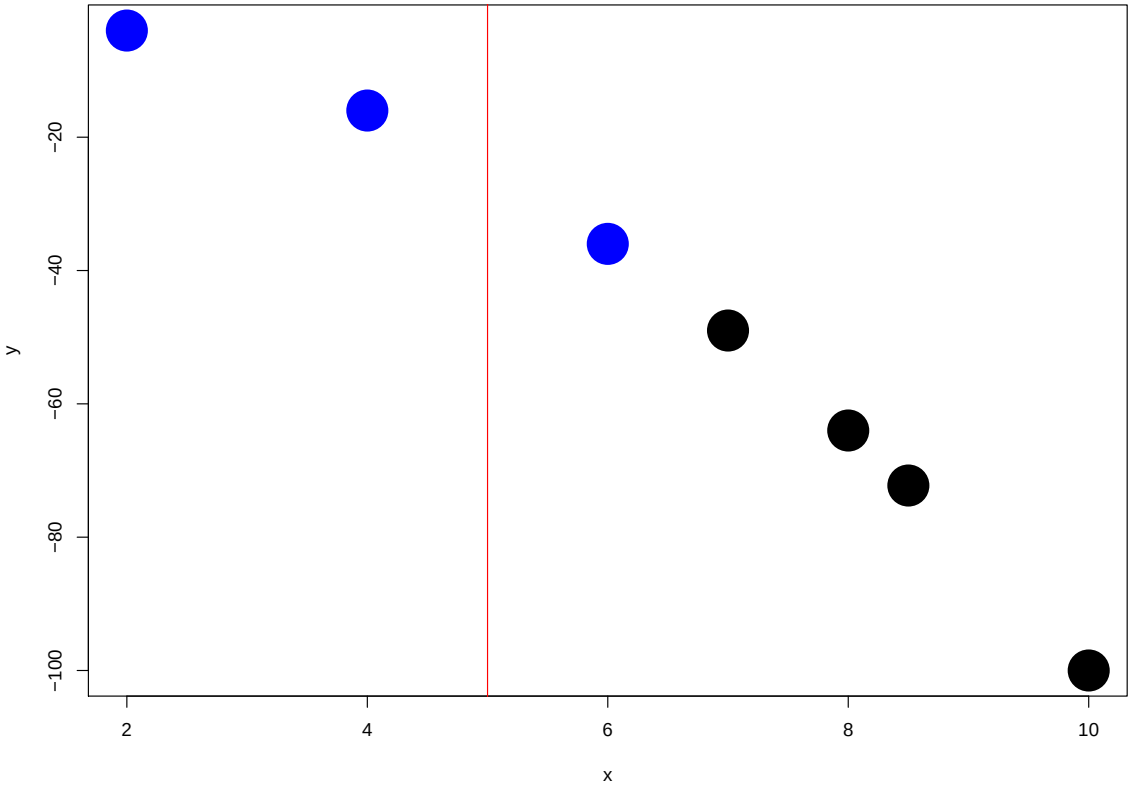
$$\hat{\boldsymbol{\beta}} \quad = \quad \min(\mathbf{y} - \mathbf{K}\boldsymbol{\beta})^\top(\mathbf{y} - \mathbf{K}\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^\top\mathbf{K}\boldsymbol{\beta}$$

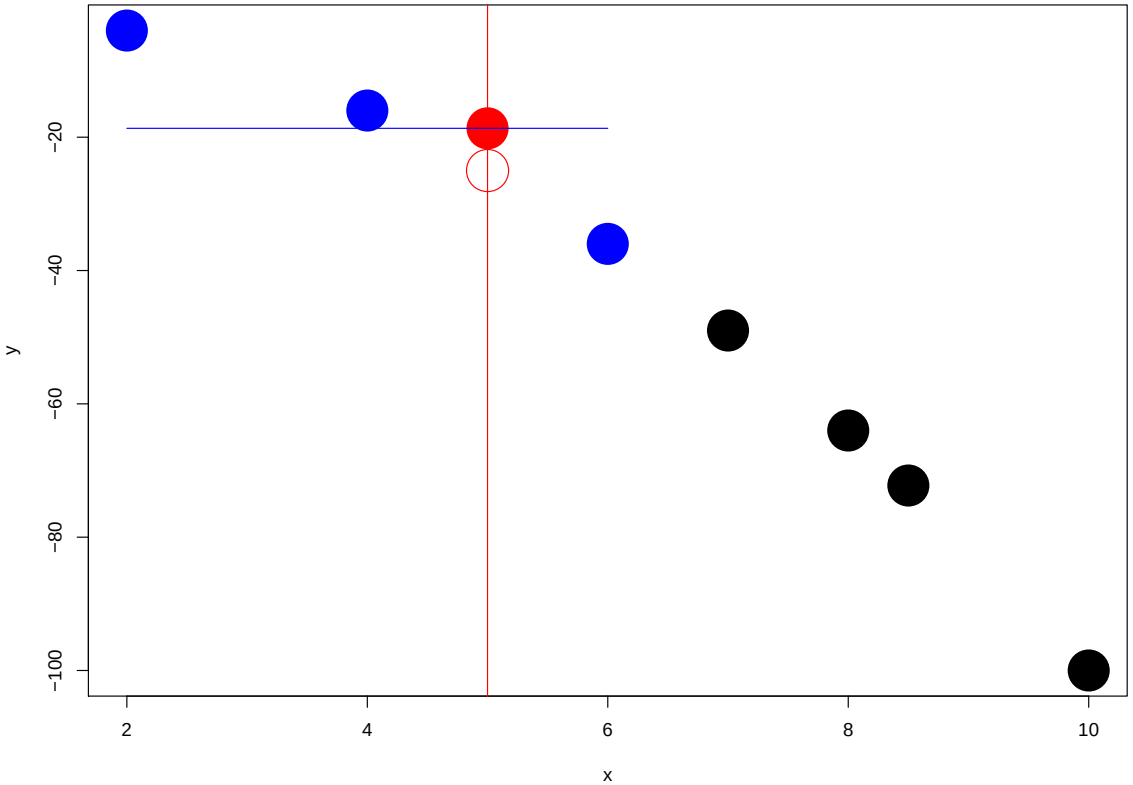
Replace $\min(\mathbf{y} - \mathbf{K}\boldsymbol{\beta})^\top(\mathbf{y} - \mathbf{K}\boldsymbol{\beta})$ with $\max(0, 1 - y_i\mathbf{K}\boldsymbol{\beta})$ to produce support vector machines.

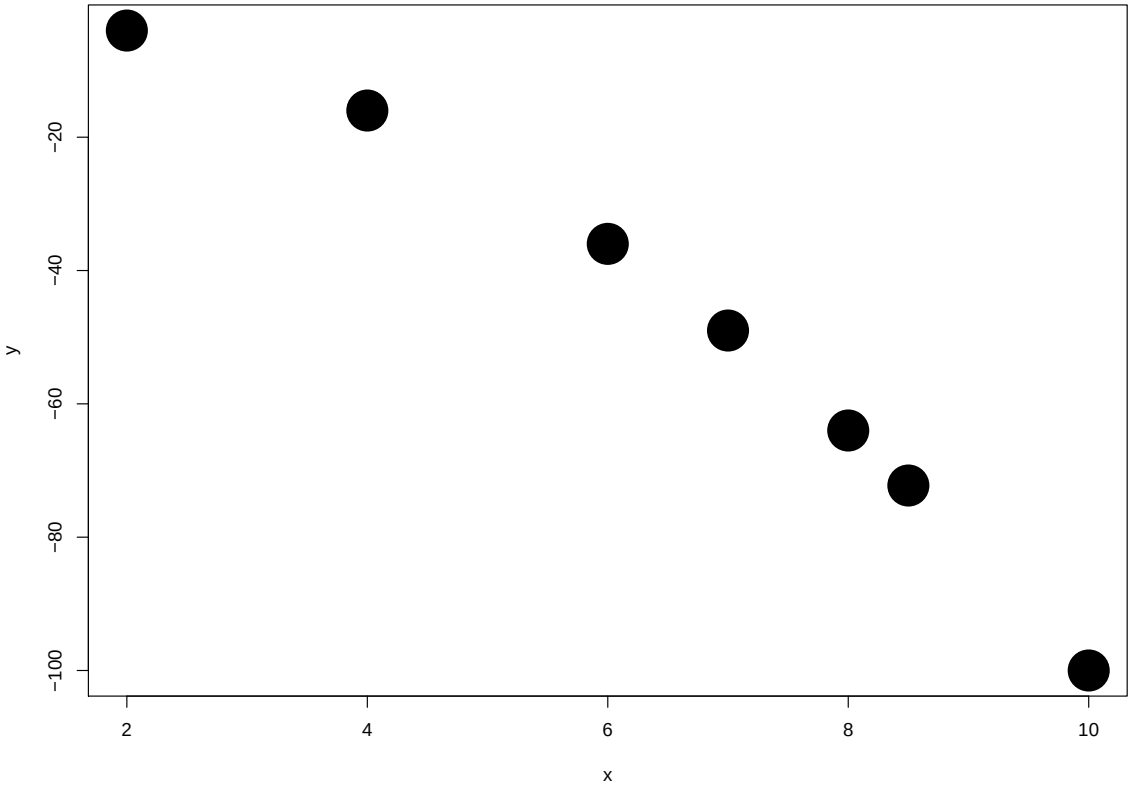
k-Nearest Neighbours: Average surrounding points

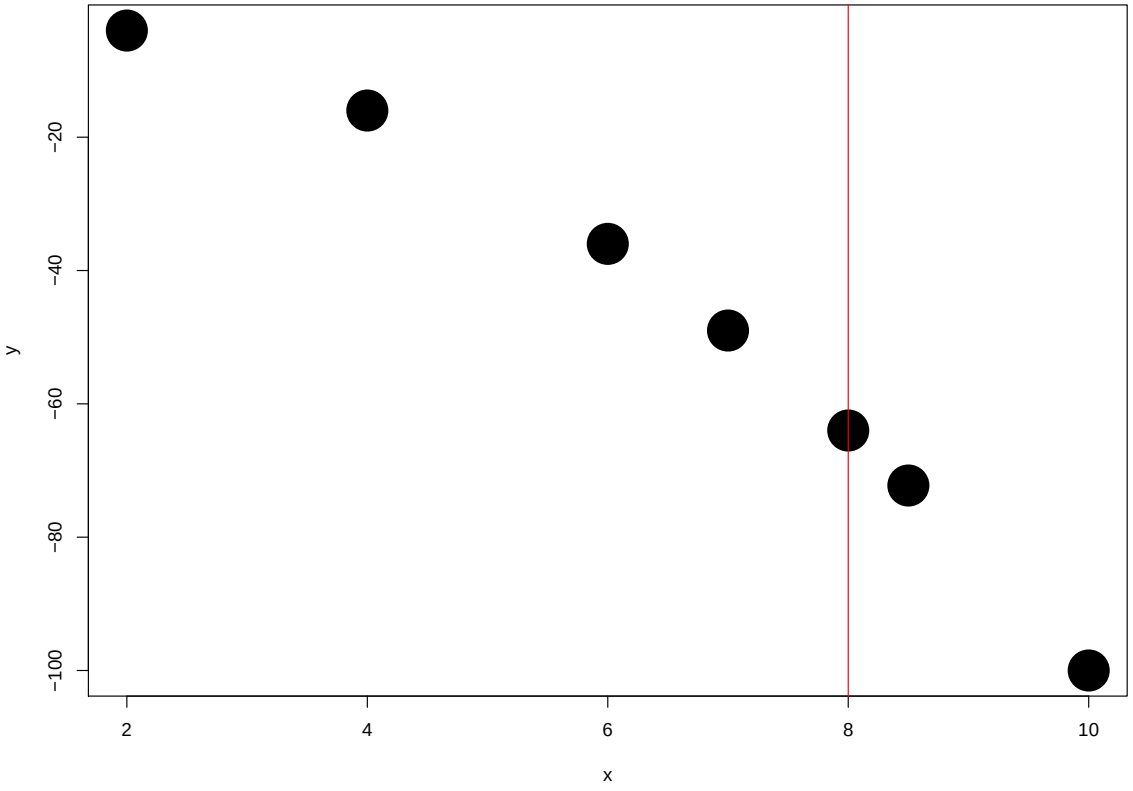

y
−20
−40
−60
−80
−100
2
4
6
8
10
x

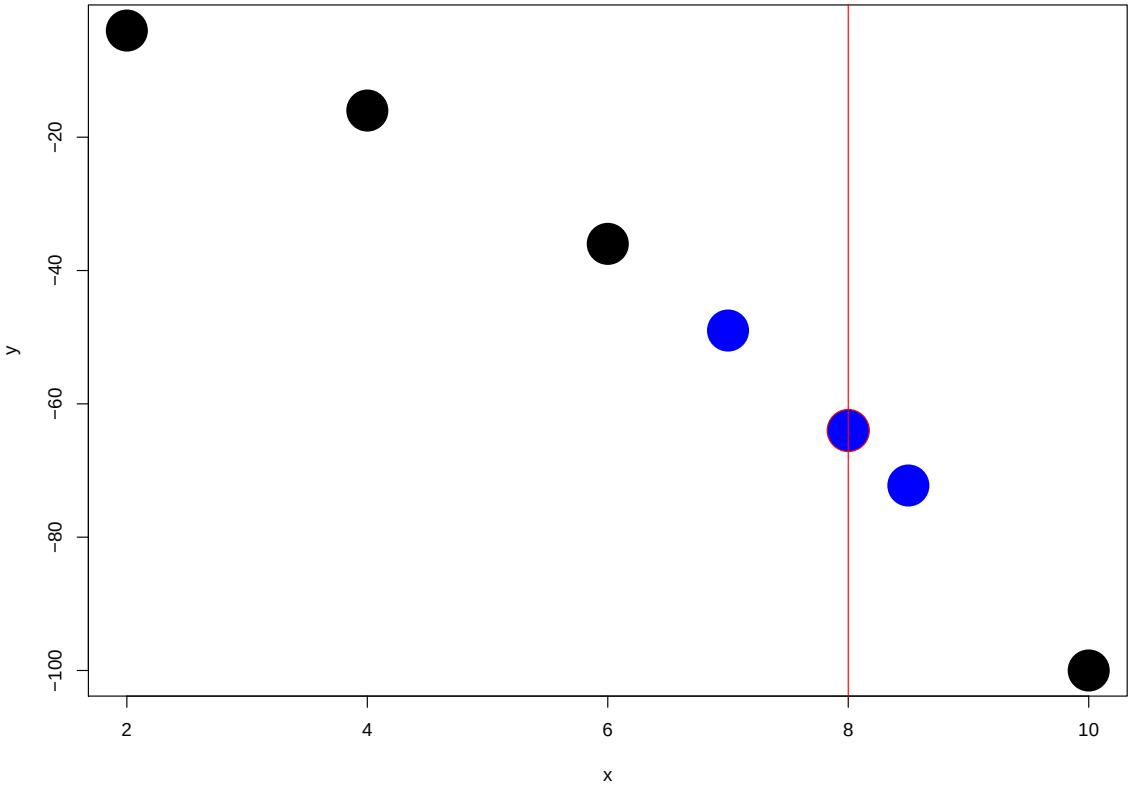

y
−20
−40
−60
−80
−100
2
4
6
8
10
x

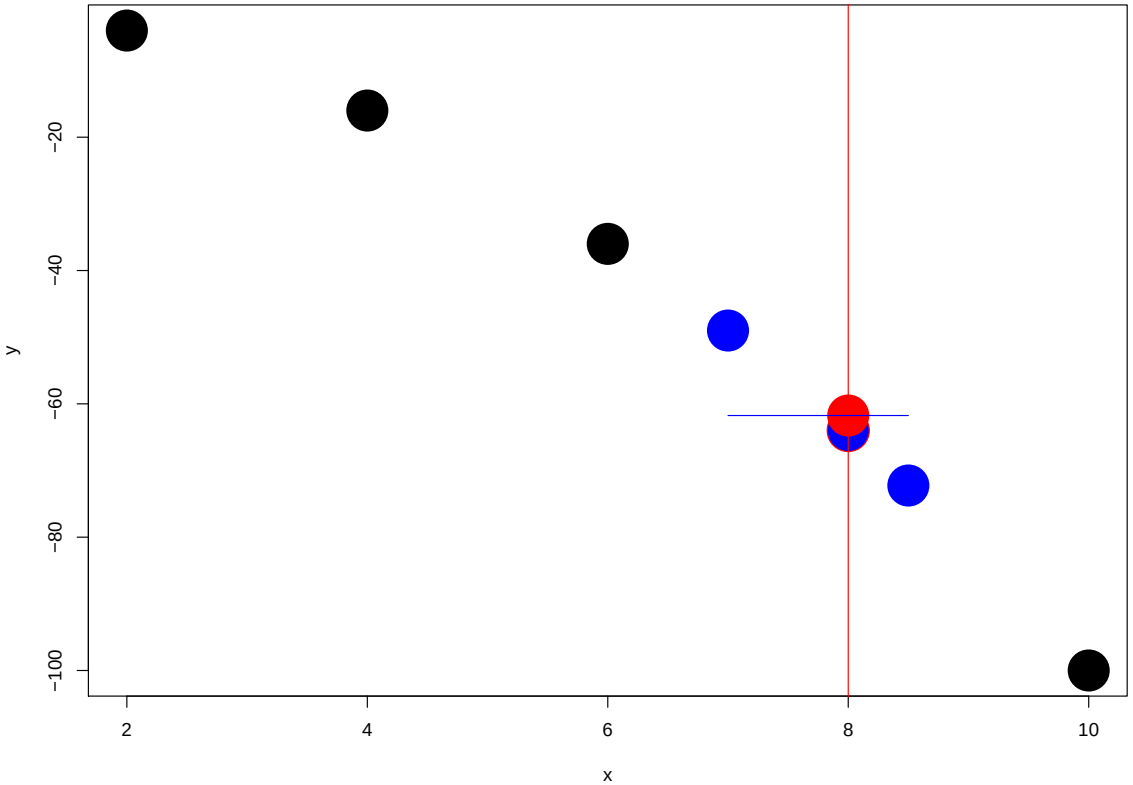

y
−20
−40
−60
−80
−100
2
4
6
8
10
x


y
−20
−40
−60
−80
−100
2
4
6
8
10
x


y
−20
−40
−60
−80
−100
2
4
6
8
10
x

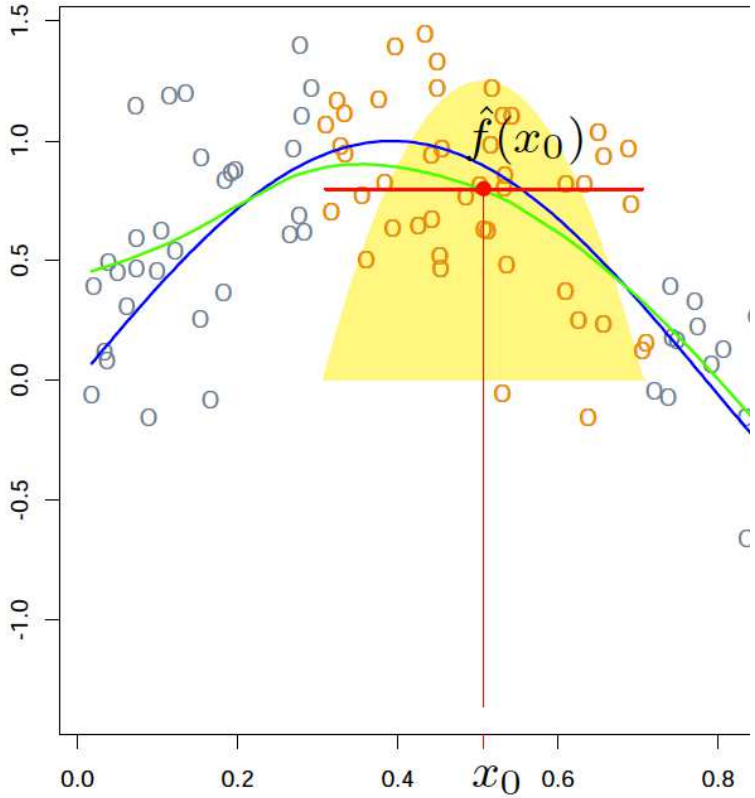

y
−20
−40
−60
−80
−100
2
4
6
8
10
x


y
−20
−40
−60
−80
−100
2
4
6
8
10
x

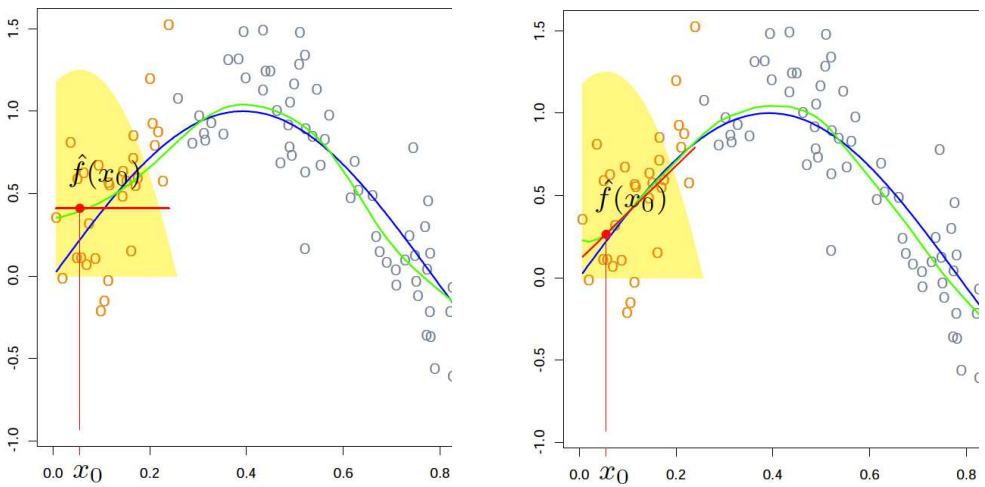

y
−20
−40
−60
−80
−100
2
4
6
8
10
x

Local Regression: Average surrounding points smoothly

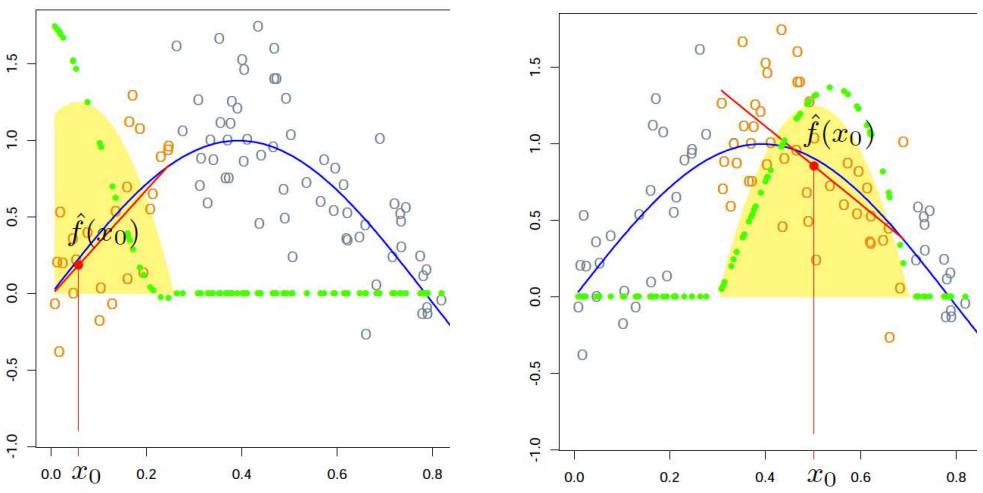
Smooth Kernel



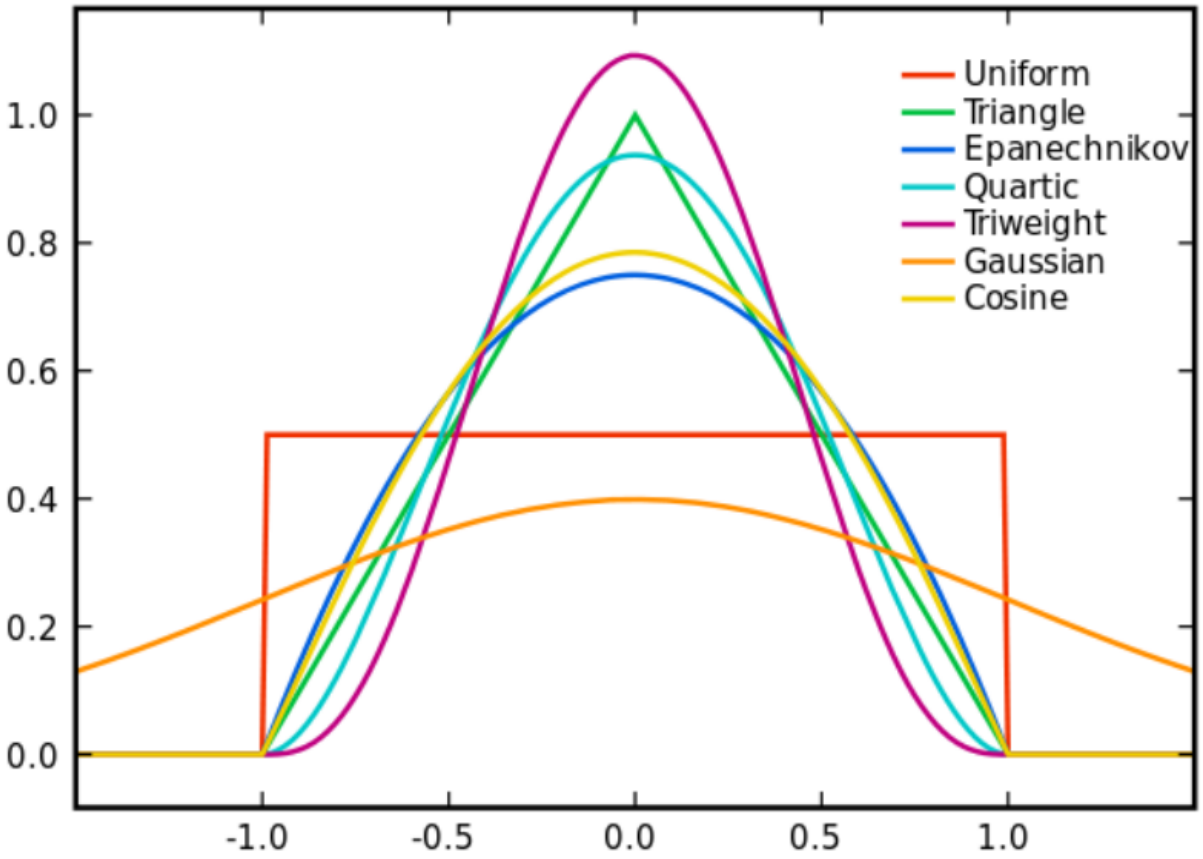
Constant versus linear



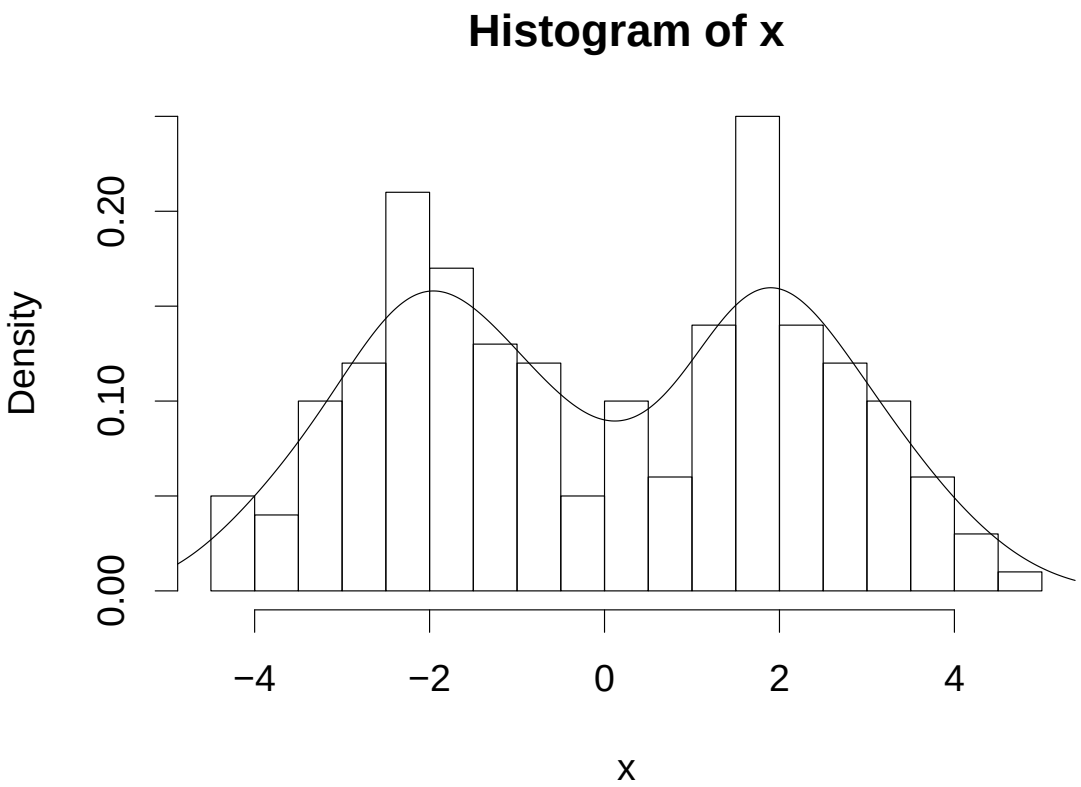
transformed kernel



Kernel functions



Smooth histograms



Polynomial logistic regression using statsmodels

```
import pandas as pd
path='data/'
filename = path+'Wage.csv'
wage = pd.read_csv(filename)
```

```
import pandas as pd
path='data/'
filename = path+'Wage.csv'
wage = pd.read_csv(filename)

from sklearn.preprocessing import PolynomialFeatures
X1 = PolynomialFeatures(1).fit_transform(wage[['age']])
X2 = PolynomialFeatures(2).fit_transform(wage[['age']])
X3 = PolynomialFeatures(3).fit_transform(wage[['age']])
X4 = PolynomialFeatures(4).fit_transform(wage[['age']])
X5 = PolynomialFeatures(5).fit_transform(wage[['age']])
```

```
import pandas as pd
path='data/'
filename = path+'Wage.csv'
wage = pd.read_csv(filename)


from sklearn.preprocessing import PolynomialFeatures
X1 = PolynomialFeatures(1).fit_transform(wage[['age']])
X2 = PolynomialFeatures(2).fit_transform(wage[['age']])
X3 = PolynomialFeatures(3).fit_transform(wage[['age']])
X4 = PolynomialFeatures(4).fit_transform(wage[['age']])
X5 = PolynomialFeatures(5).fit_transform(wage[['age']])


import statsmodels.api as sm
y = (wage['wage'] > 250).map({False:0, True:1})
lr = sm.GLM(y, X4,
        family=sm.families.Binomial(sm.families.links.logit))
lr = lr.fit()
lr.summary()
```

Result

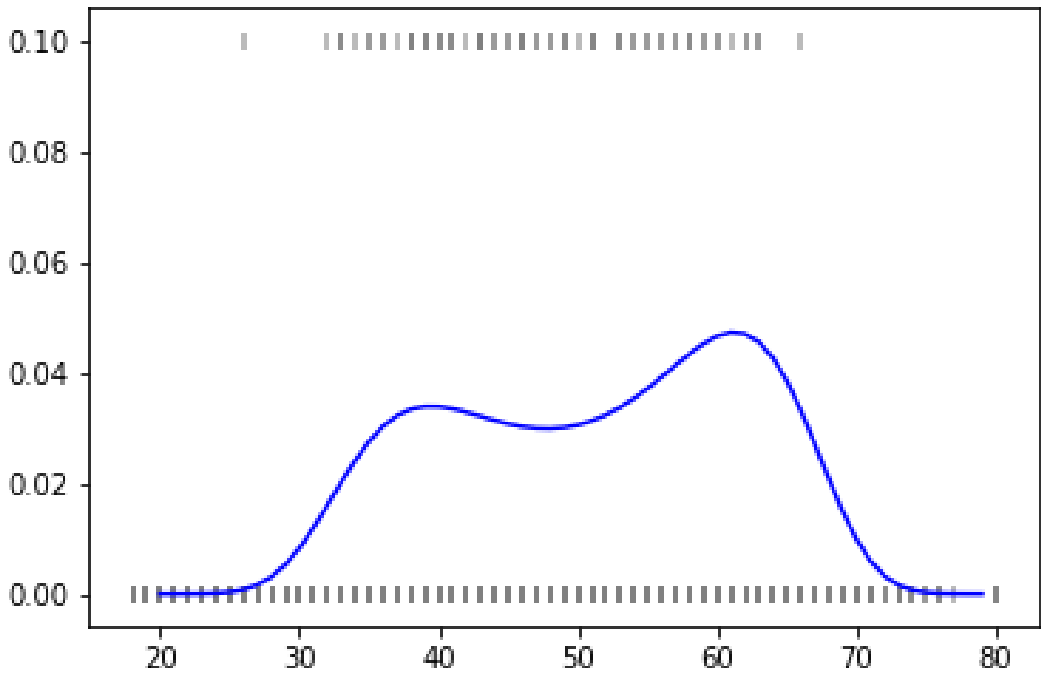
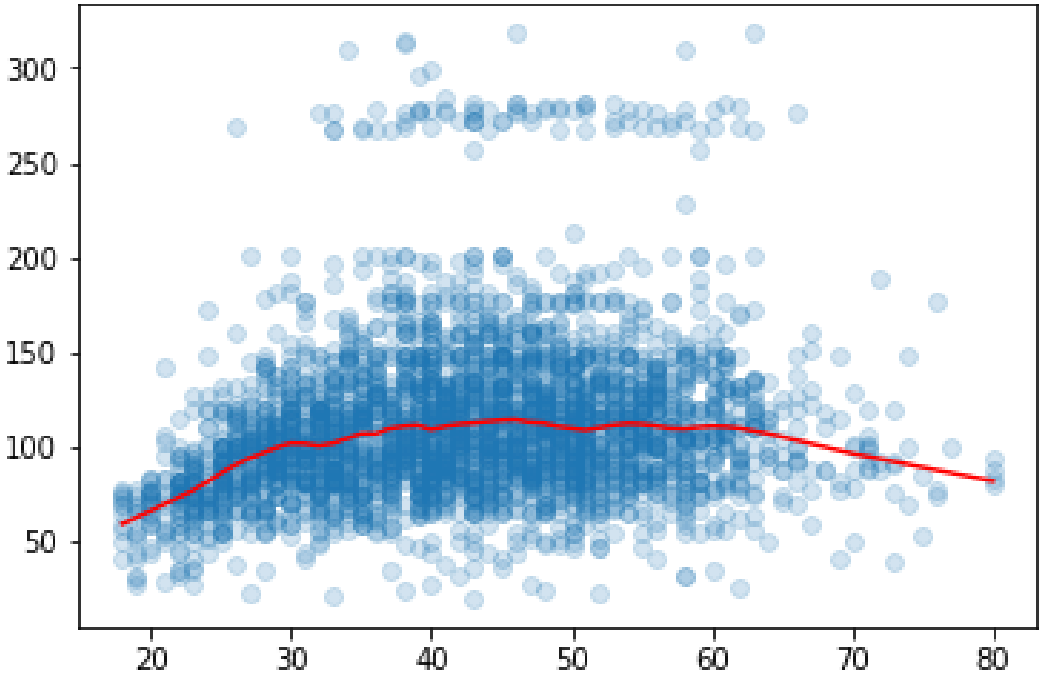
Generalized Linear Model Regression Results

Dep. Variable:	wage	**No. Observations:**	3000
Model:	GLM	**Df Residuals:**	2995
Model Family:	Binomial	**Df Model:**	4
Link Function:	logit	**Scale:**	1.0
Method:	IRLS	**Log-Likelihood:**	-350.61
Date:	Fri, 20 Apr 2018	**Deviance:**	701.22
Time:	11:11:44	**Pearson chi2:**	3.70e+03
No. Iterations:	10		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-109.5530	47.655	-2.299	0.022	-202.956	-16.150
x1	8.9950	4.187	2.148	0.032	0.789	17.201
x2	-0.2816	0.135	-2.081	0.037	-0.547	-0.016
x3	0.0039	0.002	2.022	0.043	0.000	0.008
x4	-1.949e-05	9.91e-06	-1.966	0.049	-3.89e-05	-6.41e-08

```
age_grid = np.arange(20, 80).reshape(-1,1)
X_test = PolynomialFeatures(4).fit_transform(age_grid)
y_hat = lr.predict(X_test)

import matplotlib.pyplot as plt
%matplotlib inline
```



```
from statsmodels.nonparametric.smoothers_lowess import lowess

xy = lowess(wage['wage'], wage['age'], frac=0.1)
plt.scatter(wage['age'], wage['wage'], alpha=0.2)
plt.plot(xy[:,0],xy[:,1], 'red')
```