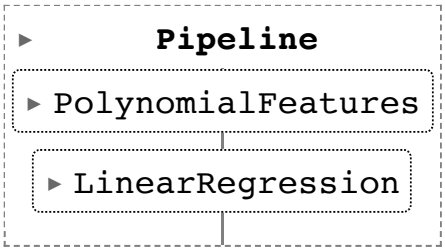
```
PyObject <class 'sklearn.pipeline.Pipeline'>
```

```
begin
    using ScikitLearn   , Random   ,DataFrames   ,Polynomials   ,Latexify
    ,Symbolics   ,CSV   ,StatsPlots
    @sk_import preprocessing : PolynomialFeatures
    @sk_import linear_model : LinearRegression
    @sk_import   pipeline : Pipeline
end
```

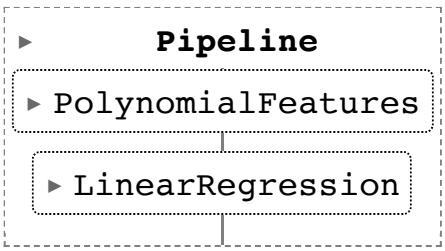
```
3×6 Matrix{Float64}:
 1.0  1.0  4.0  1.0   4.0  16.0
 1.0  2.0  5.0  4.0  10.0  25.0
 1.0  3.0  6.0  9.0  18.0  36.0
```

```
begin
    X =reshape(1:6,3, 2)

    poly = PolynomialFeatures(degree=2)
    poly.fit_transform(X)
end
```




```
begin
model = Pipeline([("poly", PolynomialFeatures(degree=3)),
("linear",LinearRegression(fit_intercept=false))])
end
```




```
begin
    @variables x1,y1
    x1 = DataFrame([1:5],:auto)
    y1 = x1[!,1].|>x->3-2x+ x^2- x^3
    res=fit!(model,([1],[2],[3],[4],[5]), y1)
end
```

```
model.named_steps["linear"].coef_|>Array.|>(d->round(d,digits=2))|>d-
>Polynomial(d,:x)|>println

```

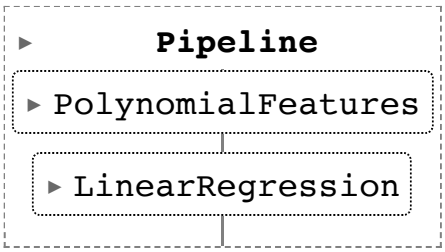
```
3.0 - 2.0*x + 1.0*x^2 - 1.0*x^3
```

2. 使用下面的的数据进行多项式回归拟合

	Y	X
1	50.4496	-2.6
2	21.2355	-1.5
3	20.4711	-1.9
4	145.15	-4.9
5	16.9187	-0.1
6	53.5485	-2.9
7	111.376	4.5
8	132.266	5.0
9	58.1507	-2.5
10	139.761	4.9
	more	
100	17.1829	-1.6

```
begin
    ## 多项式数据

    path="/Users/lunarcheung/Public/github/StatsWithJuliaBook/data/polynomialData.csv"
    data=CSV.read(path, DataFrame)
end
```

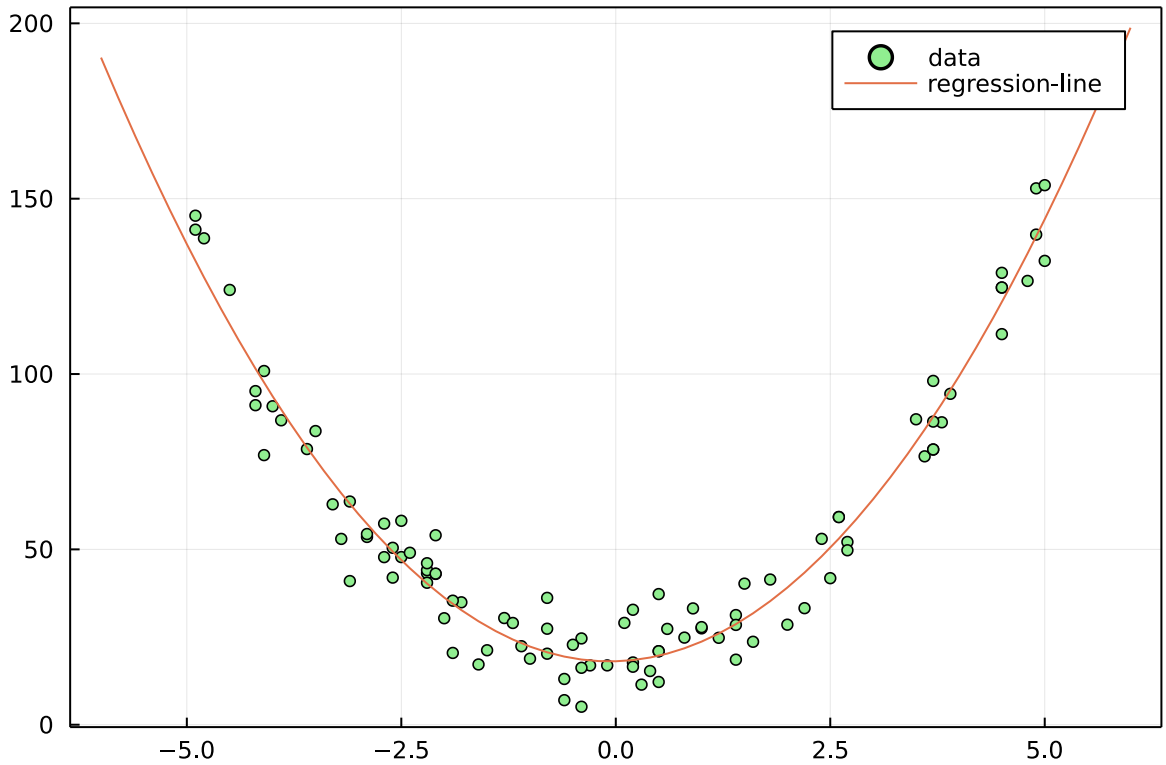



```
begin

            ##########################################################
               #模型生成能，可以改变 degree的值，生成不同阶的多项式模型#
            ##########################################################
    model2 = Pipeline([("poly", PolynomialFeatures(degree=2)),
    ("linear",LinearRegression(fit_intercept=false))])
    X2,y2=data[!,:2],data[!,:1]
    r,c=size(data)
    X2=[[X2[i]] for i in 1:r] # 预测变量稍有不同，需要生成二维数据，
    res2=fit!(model2,X2, y2)
end
```

$18.09 + 0.71 \cdot x3 + 4.9 \cdot x3^2$

```
begin
    @variables x3,y3
y3=model2.named_steps["linear"].coef_|>Array.|>(d->round(d,digits=2))|>d->Polynomial(d,:x3)
end
```




```
begin
    range=-6:0.2:6
    y_grid=[y3(x) for x in range]
    scatter(data.X,data.Y,ms=3, mc=:lightgreen,frame=:box,label="data")
    plot!(range,y_grid,label="regression-line")
end
```