9.2 Exercise: Regression

Page 142: 11-1 Suppose one of your co-workers is expecting a baby and you are participating in an office pool to predict the date of birth. Assuming that bets are placed during the 30th week of pregnancy, what variables could you use to make the best prediction? You should limit yourself to variables that are known before the birth, and likely to be available to the people in the pool.

In [2]:
```
from os.path import basename, exists


def download(url):
    filename = basename(url)
    if not exists(filename):
        from urllib.request import urlretrieve

        local, _ = urlretrieve(url, filename)
        print("Downloaded " + local)


download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/thinkstats2.py")
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/thinkplot.py")
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/nsfg.py")
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/first.py")

download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemPreg.dct")
download(
    "https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemPreg.dat.gz")
```

In [5]:
```
import numpy as np
import pandas as pd

import thinkstats2
import thinkplot

import first

live, firsts, others = first.MakeFrames()

# Variables that are statistically significate to birth weight.

import statsmodels.formula.api as smf
model = smf.ols('prglngth ~ birthord==1 + race==2 + nbrnaliv>1', data=live)
results = model.fit()
results.summary()
```

Out[5]:

OLS Regression Results

Dep. Variable:	prglngth	**R-squared:**	0.025
Model:	OLS	**Adj. R-squared:**	0.024
Method:	Least Squares	**F-statistic:**	77.25
Date:	Sun, 11 Feb 2024	**Prob (F-statistic):**	2.42e-49
Time:	17:11:04	**Log-Likelihood:**	-21960.
No. Observations:	9148	**AIC:**	4.393e+04
Df Residuals:	9144	**BIC:**	4.396e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	38.3649	0.053	718.529	0.000	38.260	38.470
birthord == 1[T.True]	0.0287	0.056	0.513	0.608	-0.081	0.138
race == 2[T.True]	0.3634	0.058	6.229	0.000	0.249	0.478
nbrnaliv > 1[T.True]	-2.9210	0.211	-13.833	0.000	-3.335	-2.507

Omnibus:	5996.608	**Durbin-Watson:**	1.650
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	132812.832
Skew:	-2.805	**Prob(JB):**	0.00
Kurtosis:	20.804	**Cond. No.**	10.0

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Page 142: 11-3 If the quantity you want to predict is a count, you can use Poisson regression, which is implemented in StatsModels with a function called poisson. It works the same way as ols and logit. As an exercise, let's use it to predict how many children a woman has born; in the NSFG dataset, this variable is called numbabes. Suppose you meet a woman who is 35 years old, black, and a college graduate whose annual household income exceed 75,000. How many children would you predict she has born?

Using a non-linear model for age.

In [13]:
```
join.numbabes.replace([97], np.nan, inplace=True)
join['age2'] = join.age_r**2
```

Using the poisson regression formula.

In [14]:
```
formula='numbabes ~ age_r + age2 + age3 + C(race) + totincr + educat'
formula='numbabes ~ age_r + age2 + C(race) + totincr + educat'
model = smf.poisson(formula, data=join)
results = model.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 1.677002
         Iterations 7
```

Out[14]:

Poisson Regression Results

Dep. Variable:	numbabes	**No. Observations:**	8884
Model:	Poisson	**Df Residuals:**	8877
Method:	MLE	**Df Model:**	6
Date:	Sun, 11 Feb 2024	**Pseudo R-squ.:**	0.03686
Time:	17:23:57	**Log-Likelihood:**	-14898.
converged:	True	**LL-Null:**	-15469.
Covariance Type:	nonrobust	**LLR p-value:**	3.681e-243

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-1.0324	0.169	-6.098	0.000	-1.364	-0.701
C(race)[T.2]	-0.1401	0.015	-9.479	0.000	-0.169	-0.111
C(race)[T.3]	-0.0991	0.025	-4.029	0.000	-0.147	-0.051
age_r	0.1556	0.010	15.006	0.000	0.135	0.176
age2	-0.0020	0.000	-13.102	0.000	-0.002	-0.002
totincr	-0.0187	0.002	-9.830	0.000	-0.022	-0.015
educat	-0.0471	0.003	-16.076	0.000	-0.053	-0.041

We find that she will have about two and half children.

In [15]:
```
columns = ['age_r', 'age2', 'age3', 'race', 'totincr', 'educat']
new = pd.DataFrame([[35, 35**2, 35**3, 1, 14, 16]], columns=columns)
results.predict(new)
```

Out[15]:
```
0    2.496802
dtype: float64
```

Page 143: 11-4 If the quantity you want to predict is categorical, you can use multinomial logistic regression, which is implemented in StatsModels with

a function called mnlogit. As an exercise, let's use it to guess whether a woman is married, cohabitating, widowed, divorced, separated, or never married; in the NSFG dataset, marital status is encoded in a variable called rmarital.Suppose you meet a woman who is 25 years old, white, and a high school graduate whose annual household income is about 45,000. What is the probability that she is married, cohabitating, etc?

Using the mnlogit function.

In [16]:
```
formula='rmarital ~ age_r + age2 + C(race) + totincr + educat'
model = smf.mnlogit(formula, data=join)
results = model.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 1.084053
         Iterations 8
```

Out[16]:

MNLogit Regression Results

Dep. Variable:	rmarital	**No. Observations:**	8884
Model:	MNLogit	**Df Residuals:**	8849
Method:	MLE	**Df Model:**	30
Date:	Sun, 11 Feb 2024	**Pseudo R-squ.:**	0.1682
Time:	18:19:44	**Log-Likelihood:**	-9630.7
converged:	True	**LL-Null:**	-11579.
Covariance Type:	nonrobust	**LLR p-value:**	0.000

rmarital=2	**coef**	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	9.0156	0.805	11.199	0.000	7.438	10.593
C(race)[T.2]	-0.9237	0.089	-10.418	0.000	-1.097	-0.750
C(race)[T.3]	-0.6179	0.136	-4.536	0.000	-0.885	-0.351
age_r	-0.3635	0.051	-7.150	0.000	-0.463	-0.264
age2	0.0048	0.001	6.103	0.000	0.003	0.006
totincr	-0.1310	0.012	-11.337	0.000	-0.154	-0.108
educat	-0.1953	0.019	-10.424	0.000	-0.232	-0.159
rmarital=3	**coef**	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	2.9570	3.020	0.979	0.328	-2.963	8.877
C(race)[T.2]	-0.4411	0.237	-1.863	0.062	-0.905	0.023
C(race)[T.3]	0.0591	0.336	0.176	0.860	-0.600	0.718
age_r	-0.3177	0.177	-1.798	0.072	-0.664	0.029
age2	0.0064	0.003	2.528	0.011	0.001	0.011
totincr	-0.3258	0.032	-10.175	0.000	-0.389	-0.263
educat	-0.0991	0.048	-2.050	0.040	-0.194	-0.004
rmarital=4	**coef**	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-3.5238	1.205	-2.924	0.003	-5.886	-1.162
C(race)[T.2]	-0.3213	0.093	-3.445	0.001	-0.504	-0.139
C(race)[T.3]	-0.7706	0.171	-4.509	0.000	-1.106	-0.436
age_r	0.1155	0.071	1.626	0.104	-0.024	0.255
age2	-0.0007	0.001	-0.701	0.483	-0.003	0.001
totincr	-0.2276	0.012	-19.621	0.000	-0.250	-0.205
educat	0.0667	0.017	3.995	0.000	0.034	0.099
rmarital=5	**coef**	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-2.8963	1.305	-2.220	0.026	-5.453	-0.339

C(race)[T.2]	-1.0407	0.104	-10.038	0.000	-1.244	-0.837
C(race)[T.3]	-0.5661	0.156	-3.635	0.000	-0.871	-0.261
age_r	0.2411	0.079	3.038	0.002	0.086	0.397
age2	-0.0035	0.001	-2.977	0.003	-0.006	-0.001
totincr	-0.2932	0.015	-20.159	0.000	-0.322	-0.265
educat	-0.0174	0.021	-0.813	0.416	-0.059	0.025
rmarital=6	**coef**	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	8.0533	0.814	9.890	0.000	6.457	9.649
C(race)[T.2]	-2.1871	0.080	-27.211	0.000	-2.345	-2.030
C(race)[T.3]	-1.9611	0.138	-14.188	0.000	-2.232	-1.690
age_r	-0.2127	0.052	-4.122	0.000	-0.314	-0.112
age2	0.0019	0.001	2.321	0.020	0.000	0.003
totincr	-0.2945	0.012	-25.320	0.000	-0.317	-0.272
educat	-0.0742	0.018	-4.169	0.000	-0.109	-0.039

In [18]:
```
columns = ['age_r', 'age2', 'race', 'totincr', 'educat']
new = pd.DataFrame([[25, 25**2, 2, 11, 12]], columns=columns)
results.predict(new)
```

Out[18]:

	0	1	2	3	4	5
0	0.750028	0.126397	0.001564	0.033403	0.021485	0.067122

0.750028 is the predicted probability of being married.

0.126397 is the predicted probability of being unmarried.

0.001564 is the predicted probability of being widowed.

0.033403 is the predicted probability of being divorced.

0.021485 is the predicted probability of being separated.

0.067122 is the predicted probability of being never married.

In [ ]: