Week 9 - Assignment 11.1

In [93]:
```
#Exercise 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
#week of pregnancy, what variables could you use to make the best prediction? You s
#limit yourself to variables that are known before the birth, and likely to be avai
#the people in the pool.
```

In [94]:
```
import numpy as np
import thinkstats2
import thinkplot
import nsfg
```

In [95]:
```
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)
```

In [96]:
```
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemPreg.dc
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemPreg.da
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemResp.dc
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemResp.da
```

In [97]:
```
#Import nsfg data
df_nsfg = nsfg.ReadFemPreg()

live = df_nsfg[df_nsfg.outcome == 1]
not_live = df_nsfg[df_nsfg.outcome != 1]

live = live[live.prglngth>30] # pregnany length over 30 weeks.
live.columns
```

Out[97]:
```
Index(['caseid', 'pregordr', 'howpreg_n', 'howpreg_p', 'moscurrp', 'nowprgdk',
       'pregend1', 'pregend2', 'nbrnaliv', 'multbrth',
       ...
       'laborfor_i', 'religion_i', 'metro_i', 'basewgt', 'adj_mod_basewgt',
       'finalwgt', 'secu_p', 'sest', 'cmintvw', 'totalwgt_lb'],
      dtype='object', length=244)
```

In [99]:
```
import statsmodels.formula.api as statsformula

#Following variables were used by the author in the solution

#NBRNALIV - How many babies did you have
#that were born alive? Please include babies that may have died
```

```
#shortly after birth and babies that you placed for adoption.

#Value Label Total
# 1 BLACK
# 2 WHITE
# 3 OTHER

#birthord == 1 -> Birth order. 1 for first birth.

model = statsformula.ols('prglngth ~ birthord==1 + race==1 + nbrnaliv>1', data=live
results = model.fit()
results.summary()
```

Out[99]:

OLS Regression Results

Dep. Variable:	prglngth	**R-squared:**	0.011
Model:	OLS	**Adj. R-squared:**	0.011
Method:	Least Squares	**F-statistic:**	33.07
Date:	Sun, 12 Feb 2023	**Prob (F-statistic):**	3.03e-21
Time:	15:32:01	**Log-Likelihood:**	-18249.
No. Observations:	8884	**AIC:**	3.651e+04
Df Residuals:	8880	**BIC:**	3.653e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	38.8835	0.031	1264.825	0.000	38.823	38.944
birthord == 1[T.True]	0.1027	0.040	2.557	0.011	0.024	0.181
race == 1[T.True]	-0.1236	0.046	-2.712	0.007	-0.213	-0.034
nbrnaliv > 1[T.True]	-1.4876	0.165	-9.042	0.000	-1.810	-1.165

Omnibus:	1579.887	**Durbin-Watson:**	1.620
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	6142.785
Skew:	-0.847	**Prob(JB):**	0.00
Kurtosis:	6.705	**Cond. No.**	9.59

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

In [100…

```
# Finding the statistically significant effect of age at pregnancy on pregnancy len
model = statsformula.ols('prglngth ~ birthord==1 + race==1 + agepreg > 25', data=li
results = model.fit()
results.summary()
```

Out[100]:

OLS Regression Results

Dep. Variable:	prglngth	**R-squared:**	0.002
Model:	OLS	**Adj. R-squared:**	0.002
Method:	Least Squares	**F-statistic:**	5.954
Date:	Sun, 12 Feb 2023	**Prob (F-statistic):**	0.000473
Time:	15:32:02	**Log-Likelihood:**	-18290.
No. Observations:	8884	**AIC:**	3.659e+04
Df Residuals:	8880	**BIC:**	3.662e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	38.8741	0.041	958.395	0.000	38.795	38.954
birthord == 1[T.True]	0.1114	0.042	2.667	0.008	0.030	0.193
race == 1[T.True]	-0.1332	0.046	-2.877	0.004	-0.224	-0.042
agepreg > 25[T.True]	-0.0319	0.042	-0.756	0.449	-0.115	0.051

Omnibus:	1611.422	**Durbin-Watson:**	1.629
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	6234.959
Skew:	-0.866	**Prob(JB):**	0.00
Kurtosis:	6.721	**Cond. No.**	3.87

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

Week 9 - Assignment 11.3

In [101…

```
#Exercise 11-3.
#If the quantity you want to predict is a count, you can use Poisson regression, wh
#implemented in StatsModels with a function called poisson. It works the same way a
#ols and logit. As an exercise, let’s use it to predict how many children a woman h
#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 exceeds $75,000. How many children would you predict she
#has born?
```

In [112…

```
resp = nsfg.ReadFemResp()
resp.index = resp.caseid
join = live.join(resp, on='caseid', rsuffix='_r') #joining nsfg preg and resp df
```

```
len(join)
```

Out[112]: 8884

In [113…
```
join.numbabes.replace([97], np.nan, inplace=True) #inplace - modify the dataframe
```

In [114…
```
formula = 'numbabes ~ age_r + C(race) + totincr + educat'
model = smf.poisson(formula, data=join)
results = model.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 1.687055
         Iterations 5
```

Out[114]:

Poisson Regression Results

Dep. Variable:	numbabes	**No. Observations:**	8884
Model:	Poisson	**Df Residuals:**	8878
Method:	MLE	**Df Model:**	5
Date:	Sun, 12 Feb 2023	**Pseudo R-squ.:**	0.03109
Time:	15:35:28	**Log-Likelihood:**	-14988.
converged:	True	**LL-Null:**	-15469.
Covariance Type:	nonrobust	**LLR p-value:**	1.106e-205

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	1.0842	0.045	23.995	0.000	0.996	1.173
C(race)[T.2]	-0.1398	0.015	-9.464	0.000	-0.169	-0.111
C(race)[T.3]	-0.0914	0.025	-3.717	0.000	-0.140	-0.043
age_r	0.0208	0.001	20.474	0.000	0.019	0.023
totincr	-0.0179	0.002	-9.442	0.000	-0.022	-0.014
educat	-0.0443	0.003	-15.139	0.000	-0.050	-0.039

In [115…
```
#Predict the number of children for a woman who is 35 years old,
#black (race=1), and a college graduate(educat=16, 4yrs) whose annual household inc

import pandas as pd
import warnings
from statsmodels.tools.sm_exceptions import ConvergenceWarning
warnings.simplefilter('ignore',  pd.errors.PerformanceWarning)

join['age_2'] = join.age_r**2

columns_df = ['age_r', 'age_2', 'age3', 'race', 'totincr', 'educat']
new_df = pd.DataFrame([[35, 35**2, 35**3, 1, 14, 16]], columns=columns_df)
results.predict(new_df)
```

Out[115]:
```
0    2.342182
dtype: float64
```

Week 9 - Assignment 11.4

In [116…
```
#Exercise 11-4.
#If the quantity you want to predict is categorical, you can use multinomial logist
#regression, which is implemented in StatsModels with a function called mnlogit. As
#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 enco
#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?
```

In [117…
```
formula='rmarital ~ age_r + age_2 + 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[117]:

MNLogit Regression Results

Dep. Variable:	rmarital	**No. Observations:**	8884
Model:	MNLogit	**Df Residuals:**	8849
Method:	MLE	**Df Model:**	30
Date:	Sun, 12 Feb 2023	**Pseudo R-squ.:**	0.1682
Time:	15:35:46	**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
age_2	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
age_2	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
age_2	-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
age_2	-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
age_2	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 [118…

```
#Prediction for a woman who is 25 years old, white, and a high school graduate
#whose annual household income is about $45,000.
#High school - 12 -  12TH GRADE
columns = ['age_r', 'age_2', 'race', 'totincr', 'educat']
new = pd.DataFrame([[25, 25**2, 2, 11, 12]], columns=columns)
results.predict(new)

# There is a 75% chance for this person is currently married,
# Around 12.6% chance of Not being married but living with opposite sex partner", e
```

Out[118]:

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