PART 2: DATA MODELING

Since the most frequent diagnosis for emergency visit between 2008 and 2016 is prediabetes among Hispanic population, the model will focus on the mortality in that spectrum.

Mortality is generally calculated within a specified year. The [study (https://dlife.com/life-expectancy-prediabetes-type1-type2-type3-diabetes/)](https://dlife.com/life-expectancy-prediabetes-type1-type2-type3-diabetes/) shows that the lifespan of patients who suffer from diabetes is reduced from 6 - 10 years, depending on the type of diabetes.

- Hence, in the ***first*** model, I did not control the year after visiting ER, except between 2008 and 2016. Running the logistic regression among low incomers, the odds ratio of being Hispanic patient, compared to the rest of the race and holding age at the fixed value, dying from prediabetes after emergency visit is high. In fact, the probability is about 80%.
- The ***second model*** restricts the death within 4 years. I chose 4 years since the study shows the life span is reduced from 6 to 10 years. Moreover, if you look at the distribution of the year difference (from the Death Date to the Visit Date), there are data points until 7 years.

In [29]:
```
#libraries here
import pandas as pd
import numpy as np
import datetime as dt
import re
import math
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style= 'white')
sns.set(style = 'whitegrid', color_codes = True)
```

In [30]:
```
#Read prediabetes patient data
prediabetes_df = pd.read_csv('../new_data/predibetes_patients.csv')
```

In [31]:
```
'''
 - Add dummy column for each race

 - Add dummy column for each income tier
'''

race_dummies = pd.get_dummies(prediabetes_df.Race, prefix = "Race").iloc
[:,:-1]
prediabetes_df = pd.concat([prediabetes_df,race_dummies], axis = 1)

income_dummies = pd.get_dummies(prediabetes_df.Income_Tier, prefix = "In
come_Tier").iloc[:,:-1]
prediabetes_df = pd.concat([prediabetes_df,income_dummies], axis = 1)
```

In [32]:
```
prediabetes_df['Intercept'] = 1.0
prediabetes_df_lower = prediabetes_df[prediabetes_df.Income_Tier == 'low
er']
model = sm.Logit(prediabetes_df_lower['Death']
                 , prediabetes_df_lower[['Race_hispanic','Age','Intercep
t']])
result = model.fit()
result.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.384719
         Iterations 7
```

Out[32]:

Model:	Logit	Pseudo R-squared:	0.219
Dependent Variable:	Death	AIC:	33.6998
Date:	2019-10-29 23:58	BIC:	38.4503
No. Observations:	36	Log-Likelihood:	-13.850
Df Model:	2	LL-Null:	-17.734
Df Residuals:	33	LLR p-value:	0.020572
Converged:	1.0000	Scale:	1.0000
No. Iterations:	7.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
Race_hispanic	1.5235	1.2860	1.1846	0.2362	-0.9971	4.0441
Age	0.0794	0.0344	2.3051	0.0212	0.0119	0.1469
Intercept	-7.7201	2.8509	-2.7080	0.0068	-13.3077	-2.1325

In [33]:
```
print ("Odds Ratio \n{}".format(np.exp(result.params)))
```

```
Odds Ratio
Race_hispanic    4.588132
Age              1.082640
Intercept        0.000444
dtype: float64
```

In [34]:
```
probability = np.exp(result.params)/ (1+np.exp(result.params))
print ("Probability \n{}".format(probability))
```

```
Probability
Race_hispanic    0.821049
Age              0.519840
Intercept        0.000444
dtype: float64
```

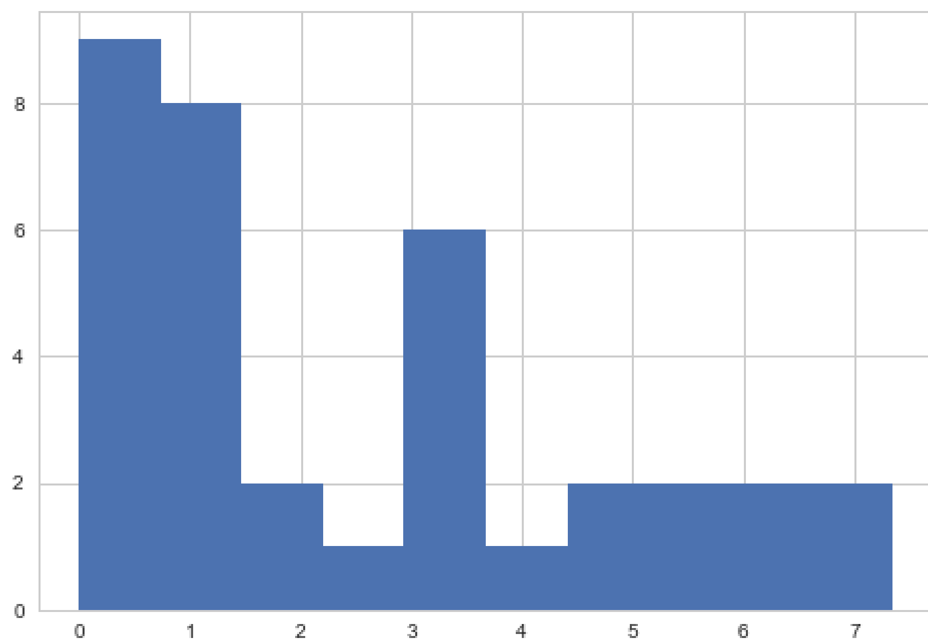
In [35]:
```
'''
- Add diff_years column to calculate the year difference between the dat
e of visit and death
'''
prediabetes_df['Date_Visit'] = pd.to_datetime(prediabetes_df['Date_Visi
t'], errors='coerce')
prediabetes_df['Death_Date'] = pd.to_datetime(prediabetes_df['Death_Dat
e'], errors='coerce')
prediabetes_df['diff_years'] = prediabetes_df['Death_Date'] - prediabete
s_df['Date_Visit']
prediabetes_df['diff_years' ]= prediabetes_df['diff_years']/np.timedelta
64(1,'Y')
```

In [36]:
```
def death_4_years (date_difference):

    if (date_difference is None) or (date_difference != date_difference
):
        return 0
    elif ( 0 <= date_difference <=4 ):
        return 1
    else:
        return 0

prediabetes_df['Death_4_years'] = prediabetes_df['diff_years'].apply(lam
bda x: death_4_years(x))
prediabetes_df.diff_years.hist()
```

Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x11d023128>



In [37]:
```
'''
- Restrict within 4 years after ER

'''

prediabetes_df['Intercept'] = 1.0
prediabetes_df_lower = prediabetes_df[prediabetes_df.Income_Tier == 'low
er']
model = sm.Logit(prediabetes_df_lower['Death_4_years']
                 , prediabetes_df_lower[['Race_hispanic','Age','Intercep
t']])
result = model.fit()
result.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.340113
         Iterations 7
```

Out[37]:

Model:	Logit	Pseudo R-squared:	0.156
Dependent Variable:	Death_4_years	AIC:	30.4881
Date:	2019-10-29 23:58	BIC:	35.2387
No. Observations:	36	Log-Likelihood:	-12.244
Df Model:	2	LL-Null:	-14.506
Df Residuals:	33	LLR p-value:	0.10416
Converged:	1.0000	Scale:	1.0000
No. Iterations:	7.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
Race_hispanic	0.8248	1.2766	0.6461	0.5182	-1.6773	3.3269
Age	0.0686	0.0365	1.8769	0.0605	-0.0030	0.1402
Intercept	-6.9028	2.9525	-2.3380	0.0194	-12.6895	-1.1160

In [38]:
```
print ("Odds Ratio \n{}".format(np.exp(result.params)))
```

```
Odds Ratio
Race_hispanic    2.281427
Age              1.070978
Intercept        0.001005
dtype: float64
```

In [39]:
```
probability = np.exp(result.params)/ (1+np.exp(result.params))
print ("Probability \n{}".format(probability))
```

```
Probability
Race_hispanic    0.695255
Age              0.517136
Intercept        0.001004
dtype: float64
```