STAT 303-2 Assignment 4 Complete

February 14, 2022

Instructions:

a. You may talk to a friend, discuss the questions and potential directions for solving them. However, you need to write your own solutions and code separately, and not as a group activity.

b. Do not write your name on the assignment. (1 point)

c. Export your Jupyter notebook as a PDF file. If you get an error, make sure you have downloaded the MikTex software (for windows) or MacTex (for mac). Note that after installing MikTex/MacTex, you will need to check for updates, install the updates if needed, and re-start your system. Submit the PDF file. (1 point)

d. Please include each question (or question number) followed by code and your answer (if applicable). Write your code in the 'Code' cells and your answer in the 'Markdown' cells of the Jupyter notebook. Ensure that the solution is well-organized, clear, and concise (3 points)

1. It's easy enough to identify different sections of the homework assignment (e.g., if there are different sections of an assignment, they're clearly distinguishable by section headers or the like)

2. It's clear which code/markdown blocks correspond to which questions.
3. There aren't excessively long outputs of extraneous information (e.g., no printouts of entire data frames without good reason)

This assignment is **due at 11:59pm on Wednesday, February 23rd**. Good luck!

Submissions will be graded with a maximum of **55 points** – 50 points for code & answers, 5 points for anonymity and proper formatting.

The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls, where bank clients were called to subscribe for a term deposit.

There is one train data - *train.csv*, which you will use to develop a model. There are two test datasets - *test1.csv* and *test2.csv*, which you will use to test your model. Each dataset has the following attributes about the clients called in the marketing campaign:

1) *age*: Age of the client

2) *education*: Education level of the client

3) *day*: Day of the month the call is made

4) *month*: Month of the call

5) *y*: did the client subscribe to a term deposit?

6) *duration*: Call duration, in seconds. This attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call *y* is obviously known. Thus, this input should only be included for inference purposes and should be discarded if the intention is to have a realistic predictive model.

(Raw data source: Source. Do not use the raw data source for this assingment. It is just for reference.)

Use *train.csv* for all questions, unless otherwise stated.

Suggestions:
(1) You may use the functions in the lecture notes for printing the confusion matrix based on test/train data.
(2) If you make variable transformations, you will need to do it for all the three datasets. Your code will be a bit concise if you make a function containing all the transformations, and then call it for the training and the two test datasets. You can put this function in the beginning of the code and keep adding transformations to it as you proceed with the assignment. You may need transformations in questions (1) and (12).

Q1

Read the datasets. Make an appropriate visualization to visualize how the proportion of clients subscribing to a term deposit change with increasing call duration.

(4 points for code)

Hints:

(a) Bin *duration* to create *duration_binned*. Group the data to find the fraction of clients responding positively to the marketing campaign for each bin in *duration_binned*. Make a lineplot of percentage of clients subscribing to a term deposit vs *duration_binned*, where the bins in *duration_binned* are arranged in increasing order of duration.

(b) You may choose an appropriate number of bins & type of binning that helps you visualize well.

(c) You may also think of other ways of visualization. You don't need to stick with this one.

[29]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

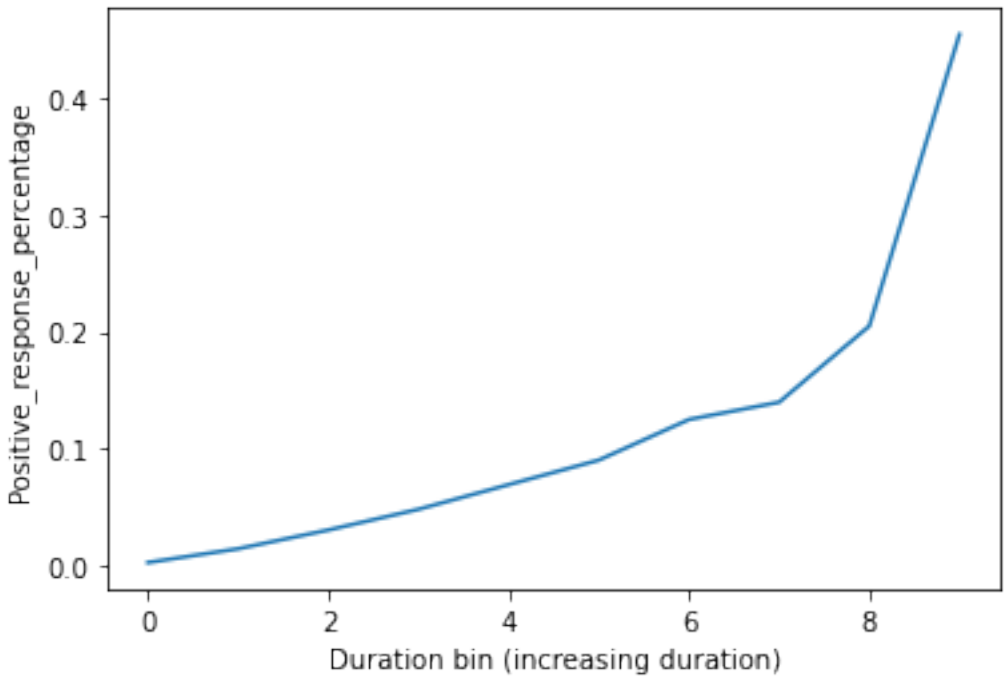
```
import statsmodels.formula.api as sm
```

[30]:
```
train = pd.read_csv('train.csv')
test1 = pd.read_csv('test1.csv')
test2 = pd.read_csv('test2.csv')
```

[31]:
```
#Note: This function contains transformation required for answering all the␣
 ↪questions
#Students don't need to explicitly have this function in their solutions.
def variable_transform(data):
    data['ynum']=0
    data.loc[data['y']=='yes','ynum'] = 1

    #Binning duration
    binned_duration = pd.qcut(train['duration'],10,retbins=True)
    bins = binned_duration[1]
    bins[0]=bins[0]-0.01
    data['duration_binned'] = pd.cut(data['duration'],bins = bins)
    dum = pd.get_dummies(data.duration_binned,drop_first = True)
    dum.columns = columns = ['duration'+str(x) for x in range(1,len(bins)-1)]
    data = pd.concat([data,dum], axis = 1)

    #Binning day
    binned_day = pd.qcut(train['day'],8,retbins=True)
    bins = binned_day[1]
    data['day_binned'] = pd.cut(data['day'],bins = bins)
    dum = pd.get_dummies(data.day_binned,drop_first = True)
    dum.columns = columns = ['day'+str(x) for x in range(1,len(bins)-1)]
    data = pd.concat([data,dum], axis = 1)
    return data

train = variable_transform(train)
test1 = variable_transform(test1)
test2 = variable_transform(test2)

#Function to jitter values in scatterplots
def jitter(values,j=0):
    return values + np.random.normal(j,0.02,values.shape)
```

[32]:
```
grouped_data = train.groupby('duration_binned')['ynum'].
 ↪agg([('Positive_response_percentage','mean'),('nobs','count')]).
 ↪reset_index(drop=False)
sns.lineplot(x = grouped_data.index, y=␣
 ↪grouped_data['Positive_response_percentage'])
plt.xlabel('Duration bin (increasing duration)')
```

[32]:
```
Text(0.5, 0, 'Duration bin (increasing duration)')
```



Q2

Based on the plot in (1), comment whether *duration* seems to be a useful variable to predict if the client will subsribe to a term deposit.

(1 point for answer)

Yes, since the proportion of people responding positively to the marketing campaign is increasing with increasing call duration, it seems to be a useful variable to predict if the client will subscribe to a term deposit.

Q3

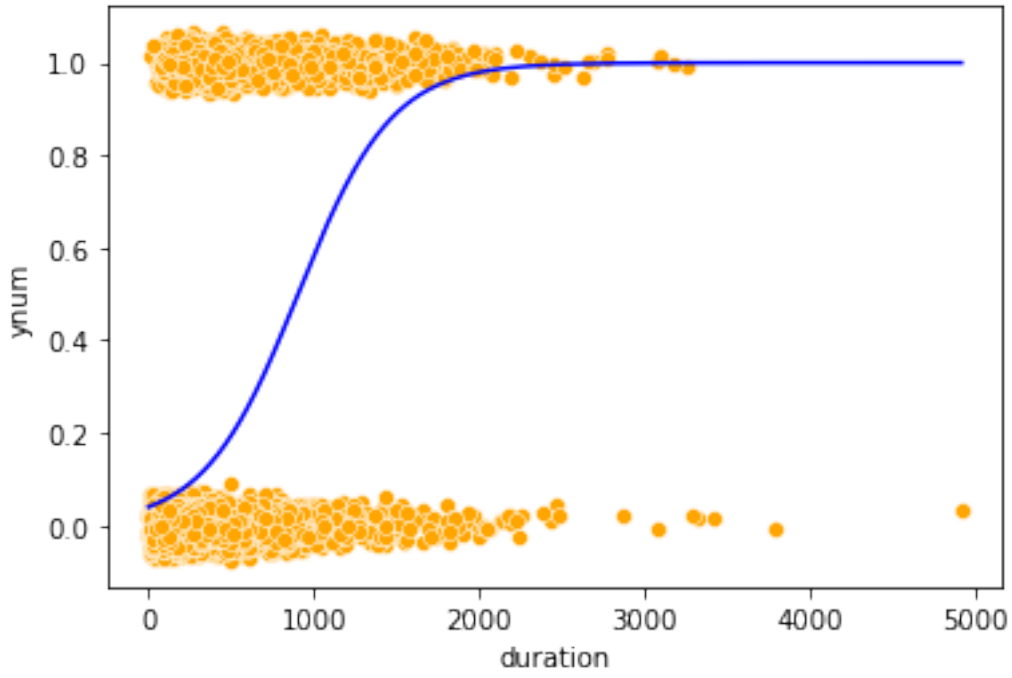
Develop a logisitic regression model to predict if the client subscribed to a term deposit based on call duration. Use the model to make a lineplot showing the probability of the client subscribing to a term deposit based on call duration.

(3 points for code)

[33]:
```
sns.scatterplot(x = jitter(train.duration), y = jitter(train.ynum), color =␣
↪'orange')
model = sm.logit(formula = 'ynum~duration', data = train).fit()
sns.lineplot(x = 'duration', y= model.predict(train), data = train, color =␣
↪'blue')
```

```
Optimization terminated successfully.
         Current function value: 0.305028
         Iterations 7
```

[33]: `<AxesSubplot:xlabel='duration', ylabel='ynum'>`



Note: Answer questions (4) to (11) based on the regression model developed in (3).

Q4

Is the regression model in statistically significant? Justify your answer.

(1 point for code, 1 point for answer)

[34]:
```
model.summary()
```

[34]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:                   ynum   No. Observations:                35000
Model:                          Logit   Df Residuals:                    34998
Method:                           MLE   Df Model:                            1
Date:                Sun, 13 Feb 2022   Pseudo R-squ.:                 0.1560
Time:                        21:53:58   Log-Likelihood:                -10676.
converged:                       True   LL-Null:                       -12650.
```

```
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -3.1841      0.030   -107.465      0.000      -3.242      -3.126
duration       0.0035   6.24e-05     56.446      0.000       0.003       0.004
==============================================================================
"""
```

Yes, the regression model is statistically significant because the p-value corresponding to the log-likelihood of the regression model is less than 5%.

Q5

What is the probability that the client subscribes to a term deposit with a 5-minute marketing call? Note that the call duration in data is given in *seconds*.

(2 points for code)

[35]:
```
model.predict(pd.DataFrame({'duration':[5*60]}))
```

[35]:
```
0    0.106491
dtype: float64
```

There is an 11% chance that the client subscribes to a term deposit with a 5-minute marketing call.

Q6

What is the minimum call duration (in minutes) for which a client has a 95% or higher chance of subscribing to a term deposit?

(3 points for code)

[36]:
```
((np.log(0.95/0.05)-model.params['Intercept'])/model.params['duration'])/60
```

[36]:
```
28.98994003859786
```

29 minutes

Q7

What is the maximum call duration (in minutes) in which a client refused to subscribe to a term deposit? What was the probability of the client subscribing to the term deposit in that call?

(3 points for code)

[37]:
```
train.loc[train.ynum==0,'duration'].max()/60
```

[37]:
```
81.96666666666667
```

Maximum call duration in which a client refused to subsribe to a term depsoit is 82 minutes

[38]:
```
model.predict(pd.DataFrame({'duration':[60*81.96666666666667]}))
```

[38]:
```
0    0.999999
dtype: float64
```

The probability of the client subscribing to a term deposit in this call was almost 100%

Q8

What is the percentage increase in the odds of a client suscribing to a term deposit when the call duration increases by a minute?

(3 points for code)

[39]:
```
np.exp(60*model.params['duration'])-1
```

[39]:
```
0.2354094217538658
```

23%

Q9

How much must the call duration increase (in minutes) so that it doubles the odds of the client subscribing to a term deposit.

(3 points for code)

[40]:
```
((np.log(2))/model.params['duration'])/60
```

[40]:
```
3.2788042131930952
```

The call duration must increase by 3.3 minutes to double the odds of the client subscribing to a term deposit.

Q10

What is minimum overall classification accuracy of the model among the classifcation accuracies on *train.csv*, *test1.csv* and *test2.csv*? Consider a threshold of 30% when classifying observations.

(3 points for code)

[41]:
```
#Function to compute confusion matrix and prediction accuracy on training data
def confusion_matrix_train(model, cutoff=0.5):
    # Confusion matrix
    cm_df = pd.DataFrame(model.pred_table(threshold=cutoff))
    #Formatting the confusion matrix
    cm_df.columns = ['Predicted 0', 'Predicted 1']
    cm_df = cm_df.rename(index={0: 'Actual 0',1: 'Actual 1'})
```

```
    cm = np.array(cm_df)
    # Calculate the accuracy
    accuracy = 100*(cm[0,0]+cm[1,1])/cm.sum()
    return cm_df, accuracy
```

[42]:
```
confusion_matrix_train(model,cutoff = 0.3)
```

[42]:
```
(          Predicted 0  Predicted 1
 Actual 0      29778.0       1118.0
 Actual 1       2896.0       1208.0,
 88.53142857142858)
```

[43]:
```
#Function to compute confusion matrix and prediction accuracy on test data
def confusion_matrix_test(data,actual_values,model,cutoff=0.5):
#Predict the values using the Logit model
    pred_values = model.predict(data)
# Specify the bins
    bins=np.array([0,cutoff,1])
#Confusion matrix
    cm = np.histogram2d(actual_values, pred_values, bins=bins)[0]
    cm_df = pd.DataFrame(cm)
    cm_df.columns = ['Predicted 0','Predicted 1']
    cm_df = cm_df.rename(index={0: 'Actual 0',1:'Actual 1'})
# Calculate the accuracy
    accuracy = 100*(cm[0,0]+cm[1,1])/cm.sum()
# Return the confusion matrix and the accuracy
    return cm_df, accuracy
```

[44]:
```
confusion_matrix_test(test1,test1.ynum, model,cutoff = 0.3)
```

[44]:
```
(          Predicted 0  Predicted 1
 Actual 0       4331.0        177.0
 Actual 1        414.0        178.0,
 88.41176470588235)
```

[45]:
```
confusion_matrix_test(test2,test2.ynum, model,cutoff = 0.3)
```

[45]:
```
(          Predicted 0  Predicted 1
 Actual 0       4355.0        163.0
 Actual 1        418.0        175.0,
 88.63236157307767)
```

The minimum overall classification accuracy is 88%.

Q11

What is maximum *false negative rate* of the model among the *false negative rates* on *train.csv*, *test1.csv* and *test2.csv*? Consider a threhold of 30% when classifying observations.

False negative rate (FNR) is the proportion of positives which yield negative test outcomes with the test, i.e., the conditional probability of a negative test result given that the condition being looked for is present (Source). Here, FNR will be the proportion of clients predicted to **not** subscribe to a term deposit among those who actually subscribed to a term deposit.

(3 points for code)

[132]:
```
np.max([2896/(1208+2896),414/(414+178),418/(418+175)])
```

[132]:
```
0.7056530214424951
```

71%

Q12

Develop a logistic regression model to predict the probability of a client subscribing to a term deposit based on *age, education* and the two-factor interaction between *age* and *education*. Based on the model, answer:

(a) People with which type of education (primary / secondary / tertiary / unknown) have the highest percentage increase in odds of subscribing to a term deposit with a unit increase in age? Justify your answer.

(b) What is the percentage increase in odds of a person subscribing to a term deposit for a unit increase in age, if the person has tertiary education.

(c) What is the percentage increase in odds of a person subscribing to a term deposit for a unit increase in age, if the person has primary education.

(1 point for developing the model, 3 points for (a), 3 points for (b), 3 points for (c))

[46]:
```
model = sm.logit(formula = 'ynum~education*age', data = train).fit()
model.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.356771
         Iterations 7
```

[46]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:                   ynum   No. Observations:                35000
Model:                          Logit   Df Residuals:                    34992
Method:                           MLE   Df Model:                            7
Date:                Sun, 13 Feb 2022   Pseudo R-squ.:                 0.01288
Time:                        21:54:45   Log-Likelihood:                -12487.
converged:                       True   LL-Null:                       -12650.
Covariance Type:            nonrobust   LLR p-value:                 1.897e-66
==============================================================================
```

```
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
--------------
Intercept                     -4.6063      0.209    -22.083      0.000
-5.015      -4.197
education[T.secondary]         2.1168      0.231      9.180      0.000
1.665       2.569
education[T.tertiary]          2.8103      0.238     11.832      0.000
2.345       3.276
education[T.unknown]           2.6776      0.359      7.455      0.000
1.974       3.382
age                            0.0462      0.004     11.636      0.000
0.038       0.054
education[T.secondary]:age    -0.0371      0.005     -8.063      0.000
-0.046      -0.028
education[T.tertiary]:age     -0.0447      0.005     -9.230      0.000
-0.054      -0.035
education[T.unknown]:age      -0.0443      0.007     -5.957      0.000
-0.059      -0.030
==============================================================================
==============
"""
```

Percentage increase in odds for an education type 'x' will be: $e^{\beta_{age}+\beta_{x:age}}$. Since $\beta_{x:age}$ is negative for secondary / tertiary / unknown education types, people with primary education will have the highest percentage increase in odds of subscribing to a term deposit with a unit increase in age.

[47]:
```
100*(np.exp(model.params['age']+model.params['education[T.tertiary]:age'])-1)
```

[47]:
```
0.14824258262073897
```

The percentage increase in odds of a person subscribing to a term deposit for a unit increase in age, if the person has tertiary education, is 0.15%

[48]:
```
100*(np.exp(model.params['age'])-1)
```

[48]:
```
4.723644883803879
```

The percentage increase in odds of a person subscribing to a term deposit for a unit increase in age, if the person has primary education, is 5%

Q13

Develop a logistic regression model to predict the probability of a client subscribing to a term deposit based on *age, education, day* and *month*. The model must have:
(a) Minimum overall classification accuracy of 75% among the classifcation accuracies on *train.csv*, *test1.csv* and *test2.csv*.

(b) Maximum false negative rate of 50% among the false negative rates on *train.csv*, *test1.csv* and *test2.csv*.

Print the (i) model summary, and the (ii) confusion matrices for all the three datasets - *train.csv*, *test1.csv* and *test2.csv*, along with the overall classification accuracies.

Note that:
(i) You cannot use *duration* as a predictor. The predictor is not useful for prediction because its value is determined after the marketing call ends. However, after the call ends, we already know whether the client responded positively or negatively. That is why we have used *duration* only for inference in the previous questions. It helped us understand the effect of the length of the call on marketing success.
(ii) It is possible to develop the model satisfying constrains (a) and (b) with just appropriate transformation(s) of the predictor(s). However, you may consider interactions if you wish.
(iii) You are free to choose any value of thershold probability for classifying observations. However, you must use the same threshold on all the three datasets.

(10 points for code)

Friendly competition for bonus points

The top 3 teams whose models satisfy constraints (a) and (b) will be awarded 3, 2, and 1 bonus point(s) based on their maximum false negative rate (FNR). These points are in addition to any other bonus points and will directly add to the total points earned in the course.

To participate in the competition:
(i) Mention your team's FNR on the spreadsheet . Keep updating it as you improve it.
(ii) One member of the team will email me the code for the model (along with the code for relevant transformations) by **9:00 pm on 22nd February**, and will acknowledge that all team members particpated. The code of the top 3 teams claiming the points will be executed, and bonus points awarded.

[49]:
```
model = sm.logit(formula = 'ynum~education+month+age+I(age**2)+' + '+'.
 ↪join(['day'+str(x) for x in range(1,8)]), data = train).fit()
print(model.summary())
print(confusion_matrix_train(model,0.125))
print(confusion_matrix_test(test1,test1.ynum,model,0.125))
print(confusion_matrix_test(test2,test2.ynum,model,0.125))
```

```
Optimization terminated successfully.
         Current function value: 0.328253
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:                   ynum   No. Observations:                35000
Model:                          Logit   Df Residuals:                    34976
Method:                           MLE   Df Model:                           23
Date:                Sun, 13 Feb 2022   Pseudo R-squ.:                0.09178
Time:                        21:55:44   Log-Likelihood:                -11489.
converged:                       True   LL-Null:                       -12650.
```

```
Covariance Type:            nonrobust   LLR p-value:                    0.000
================================================================================
==========
                             coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
----------
Intercept                  1.2885      0.210      6.139      0.000       0.877
1.700
education[T.secondary]     0.3730      0.060      6.209      0.000       0.255
0.491
education[T.tertiary]      0.6591      0.063     10.530      0.000       0.536
0.782
education[T.unknown]       0.4814      0.099      4.884      0.000       0.288
0.675
month[T.aug]              -0.8400      0.073    -11.456      0.000      -0.984
-0.696
month[T.dec]               1.0002      0.168      5.967      0.000       0.672
1.329
month[T.feb]              -0.4707      0.089     -5.303      0.000      -0.645
-0.297
month[T.jan]              -1.0133      0.120     -8.456      0.000      -1.248
-0.778
month[T.jul]              -1.0398      0.074    -14.109      0.000      -1.184
-0.895
month[T.jun]              -0.8145      0.078    -10.454      0.000      -0.967
-0.662
month[T.mar]               1.0530      0.121      8.722      0.000       0.816
1.290
month[T.may]              -1.3557      0.068    -19.872      0.000      -1.489
-1.222
month[T.nov]              -0.5518      0.086     -6.393      0.000      -0.721
-0.383
month[T.oct]               0.8082      0.103      7.873      0.000       0.607
1.009
month[T.sep]               0.8385      0.115      7.275      0.000       0.613
1.064
age                       -0.1423      0.009    -16.219      0.000      -0.159
-0.125
I(age ** 2)                0.0016   9.42e-05     16.983      0.000       0.001
0.002
day1                      -0.1354      0.073     -1.849      0.065      -0.279
0.008
day2                       0.2580      0.065      3.979      0.000       0.131
0.385
day3                       0.1760      0.073      2.411      0.016       0.033
0.319
day4                      -0.5032      0.078     -6.483      0.000      -0.655
```

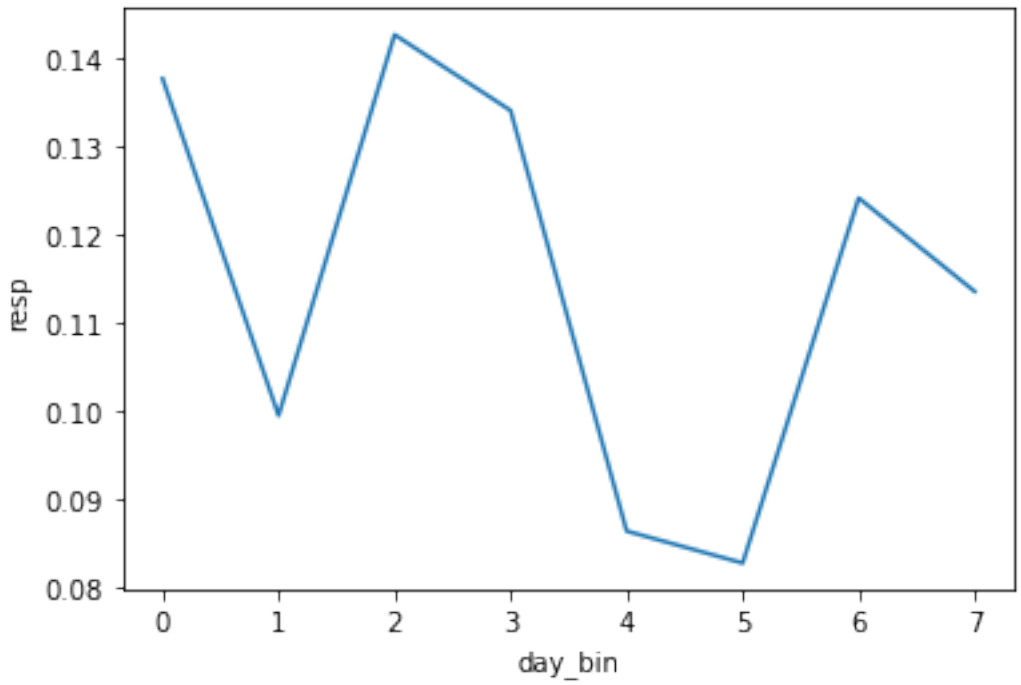
```
-0.351
day5                      -0.5465      0.084     -6.515      0.000      -0.711
-0.382
day6                       0.0365      0.069      0.531      0.595      -0.098
0.171
day7                      -0.0431      0.084     -0.514      0.607      -0.207
0.121
================================================================================
==========
(          Predicted 0  Predicted 1
Actual 0      24141.0       6755.0
Actual 1       1920.0       2184.0, 75.21428571428571)
(          Predicted 0  Predicted 1
Actual 0       3550.0        958.0
Actual 1        284.0        308.0, 75.6470588235294)
(          Predicted 0  Predicted 1
Actual 0       3522.0        996.0
Actual 1        271.0        322.0, 75.21033065936216)
```

Note: The explanation below is not required in the solution.

Following are the visualizations that indicated necessary transformations of the predictors

[50]:
```
#Binning day
binned_day = pd.qcut(train['day'],8,retbins=True)
bal_bins = binned_day[1]
train['day_binned'] = pd.cut(train['day'],bins = bal_bins)
day_data = train.groupby('day_binned')['ynum'].
 ↪agg([('resp','mean'),('nobs','count')]).reset_index(drop=False)
sns.lineplot(x = day_data.index, y= day_data['resp'])
plt.xlabel('day_bin')
```

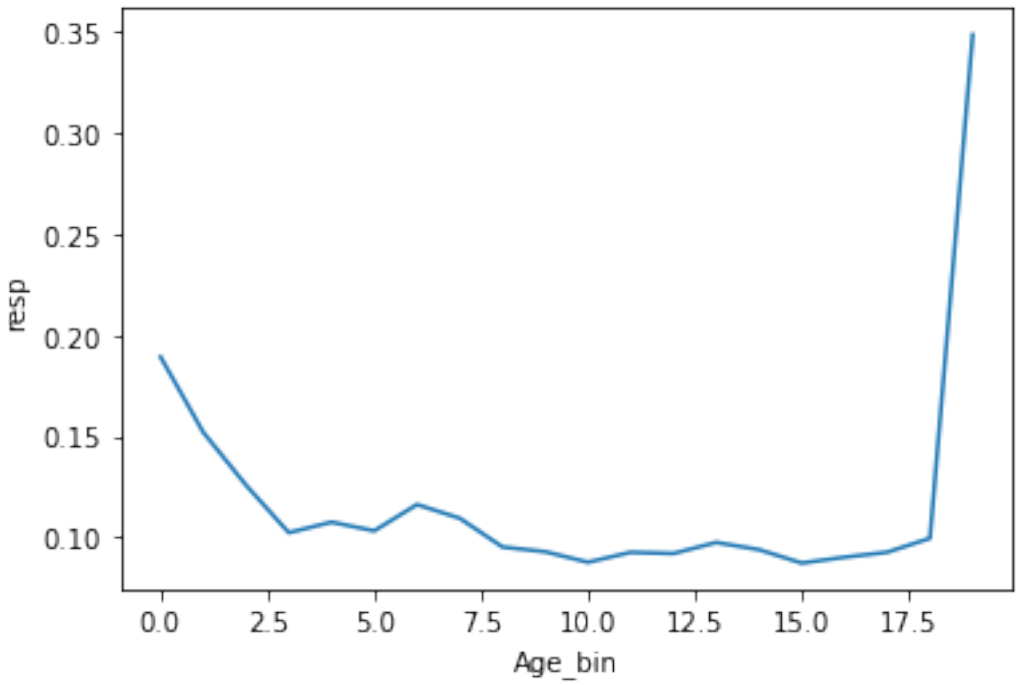
[50]:
```
Text(0.5, 0, 'day_bin')
```



Proportion of people subscribing to a term deposit vary between 8% and 14% based on *day* bins.

[51]:
```
#Binning Age
binned_age = pd.qcut(train['age'],20,retbins=True)
train['age_binned'] = binned_age[0]
age_data = train.groupby('age_binned')['ynum'].
 ↪agg([('resp','mean'),('nobs','count')]).reset_index(drop=False)
sns.lineplot(x = age_data.index, y= age_data['resp'])
plt.xlabel('Age_bin')
```

[51]:
```
Text(0.5, 0, 'Age_bin')
```

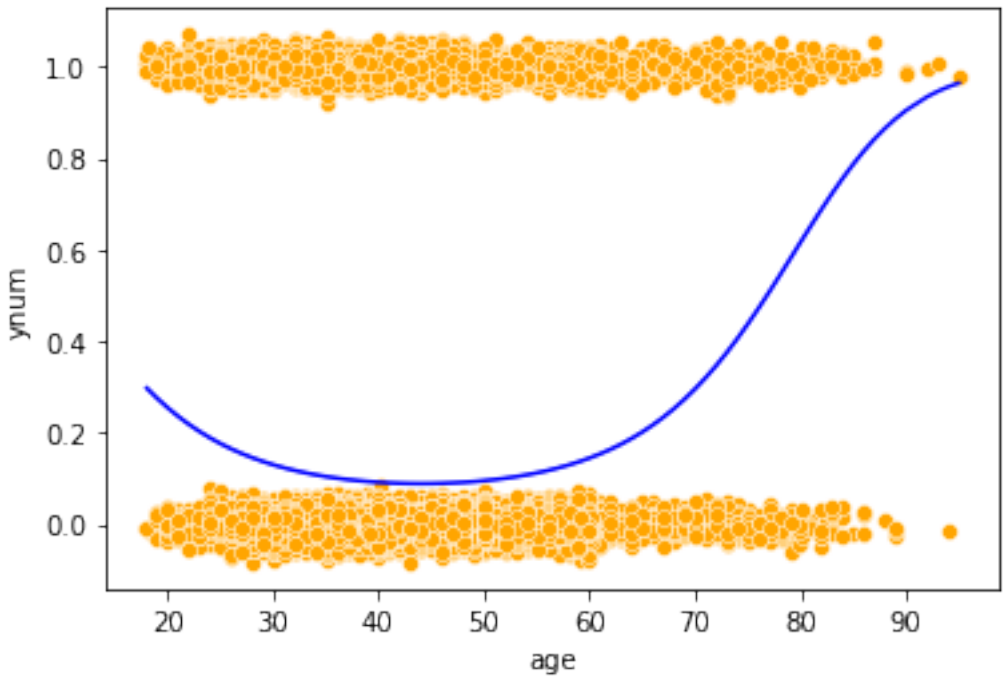


Age seems to have a quadratic relationship with the proportion of people subscribing to a term deposit

[52]:
```
#Model with the quadratic transformation of Age
def jitter(values,j):
    return values + np.random.normal(j,0.02,values.shape)
sns.scatterplot(x = jitter(train.age,0), y = jitter(train.ynum,0), data =␣
↪train, color = 'orange')
model = sm.logit(formula = 'ynum~age+I(age**2)', data = train).fit()
sns.lineplot(x = 'age', y= model.predict(train), data = train, color = 'blue')
model.llf
```

```
Optimization terminated successfully.
         Current function value: 0.352371
         Iterations 6
```

[52]:
```
-12332.996349372792
```


1.0
0.8
0.6
0.4
0.2
0.0
ynum
20
30
40
50
60
70
80
90
age