case_9.1

May 29, 2020

1 How important is the income source of an online loan applicant?

[1]:
```
import pandas as pd
import numpy  as np
from scipy.stats import chi2_contingency
import statsmodels.api as sm
from sklearn.metrics import roc_curve, auc
from sklearn.model_selection import StratifiedKFold
import matplotlib.pyplot as plt
from scipy import interp
from statsmodels.formula.api import ols
import seaborn as sns
from matplotlib.widgets import Slider, Button, RadioButtons
from scipy.optimize import fsolve

#get_ipython().magic(u'matplotlib inline')
```

1.1 Introduction (5 mts)

Business Context. Online peer-to-peer (P2P) lending has made the practice of borrowing and lending easy. In this form of lending, there is no in-person interview and a borrower can simply fill out an online form and get a loan approved. The information provided solely by a borrower is prone to exaggeration and distortion, especially when it comes to income. Every P2P lending company relies on a well-designed procedure that rejects borrowers with a high chance of not paying off their debts.

Rejecting anyone without a verified source of income is a relevant policy that lending platforms can roll out to help reduce the rate of bad loans. It is natural to suspect that if a person's income source cannot be verified, then they might default on the loan. However, from the perspective of a borrower, the verification process can be cumbersome and time-consuming and they may just switch to another platform due to this inconvenience.

Business Problem. As a data scientist at an emerging P2P lending company, you must answer the following question: **"Should the company verify the income source of an online loan applicant before approving their loan?"**

Analytical Context. The data is downloaded from LendingClub Statistics and it contains all loans issued from 2007 - 2012 along with their current loan status (fully paid or charged off). There are ~50 variables that describe the borrowers and the loans; for the sake of reducing complexity, the company has already performed a pre-screening of these variables based on existing analyses from LendingClub to select nine relevant variables, such as annual income, LendingClub credit grade, home ownership, etc. We will use a new technique, **logistic regression**, to answer our question at hand.

The case is structured as follows: you will 1) explore the existing data to get a rough idea of how each variable in the dataset interacts with the current loan status; 2) look at potential confounding effects; 3) learn the basics of logistic regression models; and finally 4) fit a series of logistic regression models to determine whether or not verifying income source is significant.

1.2 Data exploration (45 mts)

Before we begin our exploratory analysis, let's take a look at the data at our disposal:

[3]:
```
df = pd.read_csv("Lending_club_cleaned_2.csv")
df.loan_status = df.loan_status.astype(pd.api.types.
 ↪CategoricalDtype(categories=["Charged Off", "Fully Paid"]))
df.verification_status = df.verification_status.astype(pd.api.types.
 ↪CategoricalDtype(categories=['Not Verified', 'Source Verified', 'Verified']))
df.emp_length = df.emp_length.astype(pd.api.types.
 ↪CategoricalDtype(categories=["< 1 year", "1 year", "2 years", "3 years", "4␣
 ↪years", \
                                                              "5 years", "6␣
 ↪years", "7 years", "8 years", "9 years", \
                                                              "10+ years"]))
df.home_ownership = df.home_ownership.astype(pd.api.types.
 ↪CategoricalDtype(categories=["RENT", "MORTGAGE","OWN","OTHER"]))
df.term = df.term.astype(pd.api.types.CategoricalDtype(categories=[" 36␣
 ↪months", " 60 months"]))
df.grade = df.grade.astype(pd.api.types.
 ↪CategoricalDtype(categories=["A","B","C","D","E","F","G"]))
df.int_rate = df.int_rate.str.rstrip("%").astype("float")
```

[4]:
```
df.shape
```

[4]:
```
(38705, 10)
```

[5]:
```
df.head(10)
```

[5]:
```
   loan_status  annual_inc verification_status emp_length home_ownership  \
0   Fully Paid     24000.0            Verified  10+ years           RENT
1  Charged Off     30000.0     Source Verified   < 1 year           RENT
2   Fully Paid     12252.0        Not Verified  10+ years           RENT
3   Fully Paid     49200.0     Source Verified  10+ years           RENT
4   Fully Paid     80000.0     Source Verified     1 year           RENT
5   Fully Paid     36000.0     Source Verified    3 years           RENT
```

```
6   Fully Paid     47004.0        Not Verified    8 years           RENT
7   Fully Paid     48000.0     Source Verified    9 years           RENT
8  Charged Off     40000.0     Source Verified    4 years            OWN
9  Charged Off     15000.0            Verified   < 1 year           RENT

   int_rate  loan_amnt             purpose        term grade
0     10.65       5000         credit_card   36 months     B
1     15.27       2500                 car   60 months     C
2     15.96       2400      small_business   36 months     C
3     13.49      10000               other   36 months     C
4     12.69       3000               other   60 months     B
5      7.90       5000             wedding   36 months     A
6     15.96       7000  debt_consolidation   60 months     C
7     18.64       3000                 car   36 months     E
8     21.28       5600      small_business   60 months     F
9     12.69       5375               other   60 months     B
```

We have 38706 records of past transactions in this dataset. Each record corresponds to an approved loan. The first column indicates whether the borrower paid off the loan (fully paid) or not (charged off). Descriptions of the other nine columns are as follows:

annual_inc	The self-reported annual income provided by the borrower during registration.
verification_status	Indicates if income was verified by LC, not verified, or if the income source was verified
emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
home_ownership	The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: RENT, OWN, MORTGAGE, OTHER
int_rate	Interest Rate on the loan
loan_amnt	The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.
purpose	A category provided by the borrower for the loan request.
term	The number of payments on the loan. Values are in months and can be either 36 or 60.
grade	LC assigned loan grade

1.2.1 Relationships between `loan_status` and other variables (25 mts)

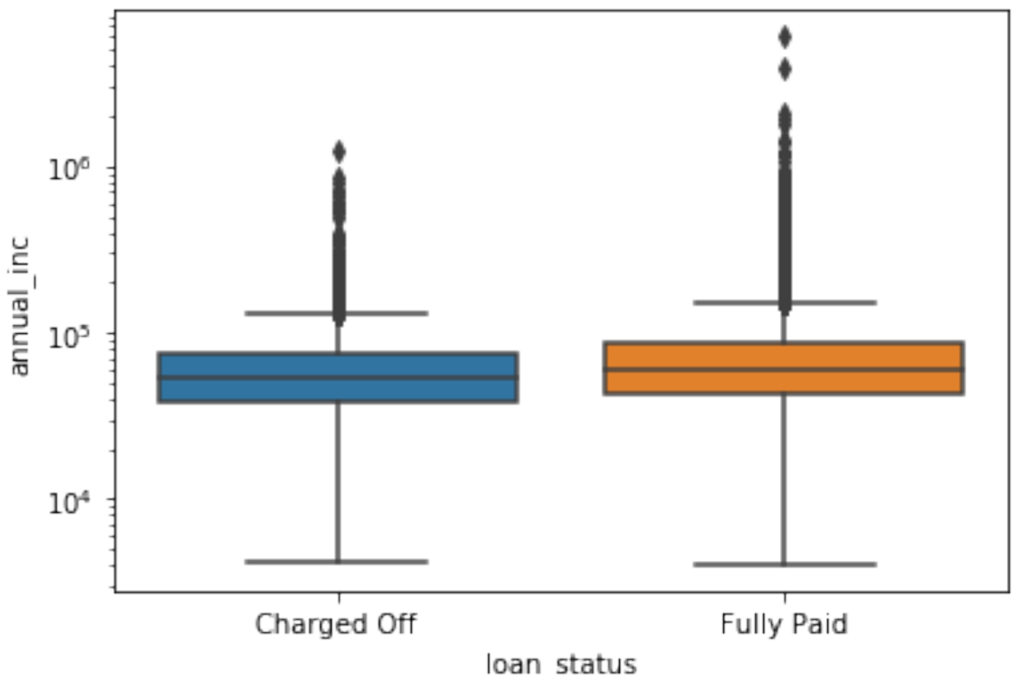
We first perform EDA to examine the pairwise relationship between `loan_status` and each of other variables. We have both discrete and continuous variables in the dataset. Let's consider the continuous variables first.

1.2.2 Exercise 1: (5 mts)

Use the `boxplot()` function from the `seaborn` package to visualize the distributions of `annual_inc`, `int_rate` and `loan_amnt` in users that fully paid their debts vs. those that didn't. Do you see a large difference in the distributions?

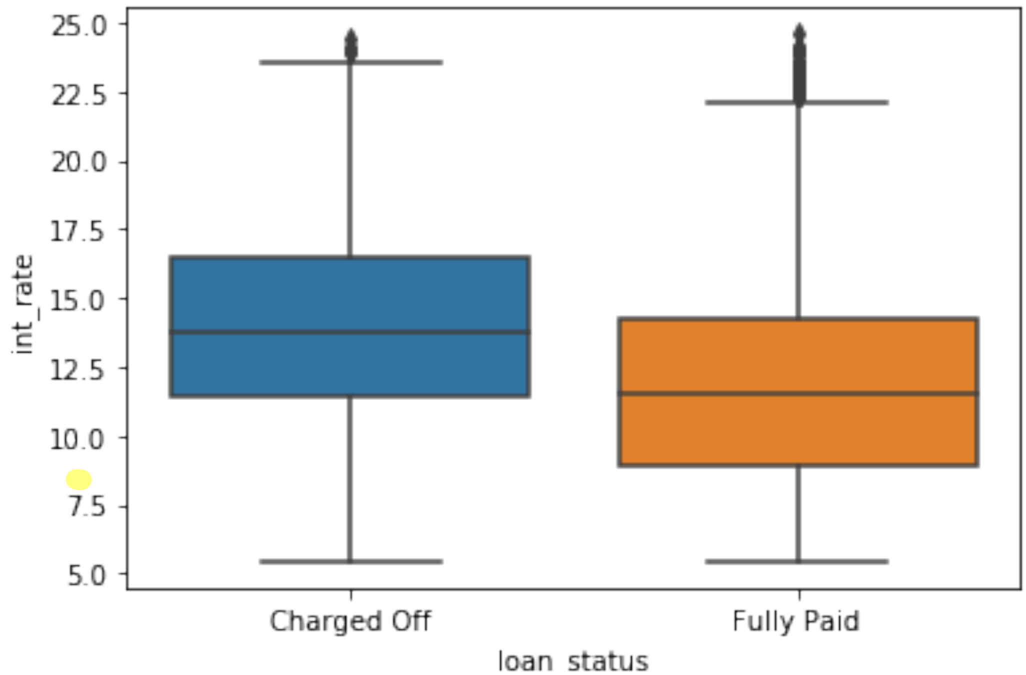
Answer. One possible solution is shown below:

[6]:
```
sns.boxplot(y="annual_inc", x="loan_status", data = df).set_yscale('log');
```



We can see a slight increase in the mean of annual income when comparing fully paid users to charged off users.

[7]:
```
sns.boxplot(y="int_rate", x ='loan_status', data = df);
```

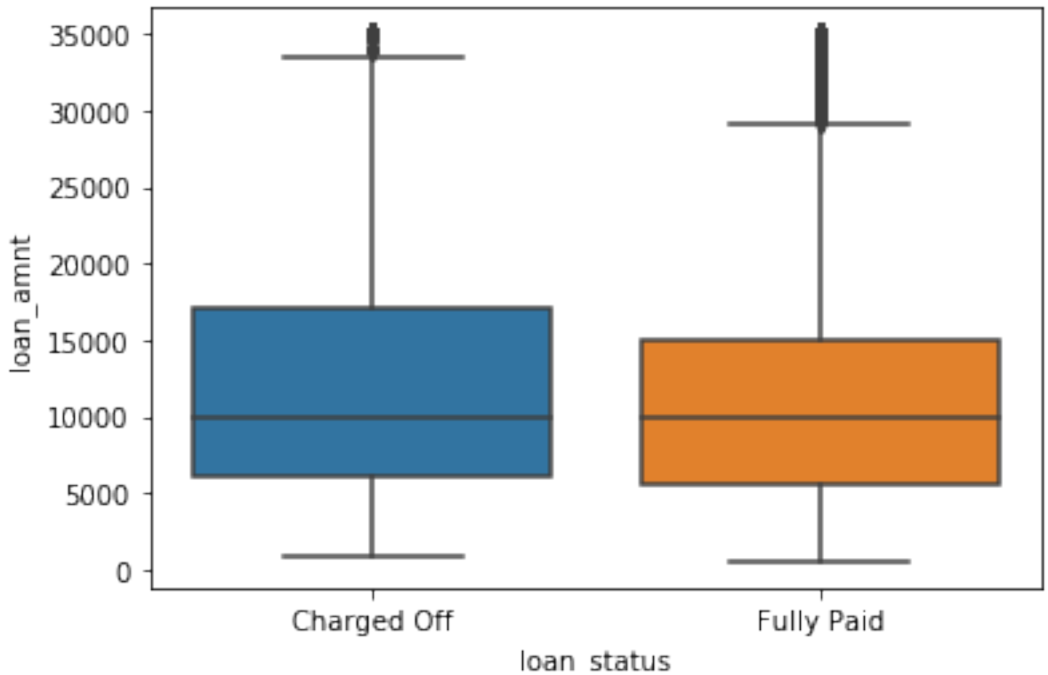


interes

The difference between charged-off users and full-paid users in the distribution of `int_rate` is more drastic. The boxplot for fully-paid users is lower than the one for charged-off users. This result is as expected as higher interest rates would increase the actual amount of money a user have to pay.

[8]:
```
sns.boxplot(y="loan_amnt", x ='loan_status', data = df)
```

[8]: <matplotlib.axes._subplots.AxesSubplot at 0x7fb99903df60>



We don't see a clear change in the distribution of `loan_amnt`. Users who fully paid their debts seem to have slightly lower amount of loans on average.

We learned in previous hypothesis testing cases that we can use contingency tables to examine the relationship between two discrete variables. A chi-square test can be performed based on the contigency table to verify if the observed relationship is statistically significant. Let's examine all discrete variables in our dataset.

1.2.3 Exercise 2: (10 mts)

Write a function to generate the contingency table between `loan_status` and another variable called `var`. Make sure to include a column for the proportion of fully-paid users in each level of `var`. Use this function to generate the contingency tables to examine the relationships between `loan_status` and all discrete variables in the dataset.

[10]:
```
def get_ct(df, var):
    ct_res = pd.crosstab(df[var], df['loan_status'], margins=True)
    ct_res["Fully Paid(%)"] = round( ct_res["Fully Paid"]/ct_res["All"]*100, 2)
    return ct_res.drop(columns="All")
```

[11]:
```
get_ct( df, 'verification_status')
```

[11]:
```
loan_status          Charged Off  Fully Paid  Fully Paid(%)
verification_status
Not Verified                2050       14449          87.58
```

```
Source Verified              1413        8406            85.61
Verified                     1977       10410            84.04
All                          5440       33265            85.94
```

[12]:
```
get_ct( df, 'emp_length' )
```

[12]:
```
loan_status  Charged Off  Fully Paid  Fully Paid(%)
emp_length
< 1 year             641        3947          86.03
1 year               460        2787          85.83
2 years              571        3823          87.01
3 years              558        3540          86.38
4 years              466        2978          86.47
5 years              459        2826          86.03
6 years              309        1922          86.15
7 years              263        1512          85.18
8 years              206        1279          86.13
9 years              159        1100          87.37
10+ years           1348        7551          84.85
All                 5440       33265          85.94
```

[13]:
```
get_ct( df, 'home_ownership' )
```

[13]:
```
loan_status     Charged Off  Fully Paid  Fully Paid(%)
home_ownership
RENT                   2752       15755          85.13
MORTGAGE               2268       14987          86.86
OWN                     402        2443          85.87
OTHER                    18          80          81.63
All                    5440       33265          85.94
```

[14]:
```
get_ct( df, 'purpose' )
```

[14]:
```
loan_status         Charged Off  Fully Paid  Fully Paid(%)
purpose
car                         155        1344          89.66
credit_card                 516        4491          89.69
debt_consolidation         2703       15572          85.21
educational                  52         265          83.60
home_improvement            333        2554          88.47
house                        59         310          84.01
major_purchase              211        1905          90.03
medical                     101         569          84.93
moving                       84         475          84.97
other                       600        3238          84.37
renewable_energy             18          77          81.05
small_business              466        1320          73.91
```

```
vacation                           50         302            85.80
wedding                            92         843            90.16
All                              5440       33265            85.94
```

[15]:
```python
get_ct( df, 'term' )
```

[15]:

loan_status	Charged Off	Fully Paid	Fully Paid(%)
term			
36 months	3073	25211	89.14
60 months	2367	8054	77.29
All	5440	33265	85.94

[16]:
```python
get_ct( df, 'grade' )
```

[16]:

loan_status	Charged Off	Fully Paid	Fully Paid(%)
grade			
A	558	9136	94.24
B	1362	10335	88.36
C	1296	6642	83.67
D	1097	4116	78.96
E	710	2101	74.74
F	317	718	69.37
G	100	217	68.45
All	5440	33265	85.94

1.2.4 Exercise 3: (5 mts)

Based on the contingency tables above, which variables seem to have associations with `loan_status`? Use chi-square test to verify if the observed associations are statistically significant.

Answer. We compute the chi-square test on each contingency table from Exercise 2:

[17]:
```python
chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['verification_status'],␣
 ↪df['loan_status']))
print('verification_status: p-value of chisquare test =', p)
```

```
verification_status: p-value of chisquare test = 6.884998281535999e-17
```

[18]:
```python
chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['emp_length'],␣
 ↪df['loan_status']))
print('emp_length: p-value of chisquare test =', p)
```

```
emp_length: p-value of chisquare test = 0.06419956286286349
```

No es estadísticamente significativo

[19]:
```python
chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['home_ownership'],␣
 ↪df['loan_status']))
```

```
print('home_ownership: p-value of chisquare test =', p)
```

```
home_ownership: p-value of chisquare test = 3.094754275730736e-05
```

[20]:
```
chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['purpose'],␣
↪df['loan_status']))
print('purpose: p-value of chisquare test =', p)
```

```
purpose: p-value of chisquare test = 6.53325963877868e-71
```

[21]:
```
chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['term'], df['loan_status']))
print('term: p-value of chisquare test =', p)
```

```
term: p-value of chisquare test = 2.7851291583847273e-194
```

[22]:
```
chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['grade'], df['loan_status']))
print('grade: p-value of chisquare test =', p)
```

```
grade: p-value of chisquare test = 0.0
```

It seems that all variables considered above are associated with `loan_status` since the proportions of fully-paid users are not the same across levels of these variables. Notably, the variable that we care about the most, `verification_status`, seems to have an counter-intuitive association with `loan_status`: the less reliable the income information is, the more likely a user would fully repay the debt, as reflected by the decreasing trend of fully paid proportions when `verification_status` increases from not verified to fully verified. The chi-square tests verify that all these observed associations, except for `emp_length`, are indeed statistically significant.

1.2.5 Exercise 4: (5 mts)

Based on the results above, which of the following statements are correct? Select all that apply.

1. There is a decrease of 2.5 on average in interest rate in users who fully paid their debts compared to users who didn't.
2. There is no statistically significant association between `emp_length` and `loan_status` since the proportion of fully paid users remains relatively the same across different levels of `emp_length`.
3. Since the chi-square test for the contingency table of `verification_status` vs. `loan_status` is significant, the decreasing trend of probability of pay-off is not likely due to randomness.
4. Since there is a significant difference between the probability of full payment in users with no verified income and users with verified income based on the chi-square test, we should always enforce income verification.

Answer. 1 is correct as indicated by the boxplots. 2 is not correct since the conclusion should be drawn based on the chi-square test (which is not signficant). 3 is not correct. The chi-sqaure test is not a trend test, it only suggests that there is significant difference in the probability of pay-off among the three levels of `verification_status`, but it cannot further verify that the differences form a specific trend. 4 is not correct since it implies a causal relationship between `verification_status` and `loan_status`. There is not enough evidence to deduce this, and we should first assess for potential confounding variables like we did in previous cases before making such a strong conclusion.

1.2.6 Stratified contigency tables of `verification_status` vs. `loan_status` (15 mts)

As we saw in the previous exercise, we need to adjust for other variables in order to remove potential confounding impacts. If we want to know if `verification_status` really is associated with `loan_status` as indicated by the contigency table above, we should consider stratifying the contingency table between `loan_status` and `verification_status` by other variables that are also associated with `loan_status`.

1.2.7 Exercise 5: (10 mts)

Write a function to generate a contingency table for `verification_status` vs. `loan_status` stratified by `stra_var`. (Hint: If `index` includes two variables, the contingency table will be stratified by the first variable). Make sure to include a column for the proportion of fully-paid users in each level of `verification_status`. Use this function to examine the relationship between `verification_status` and `loan_status` when adjusting for `home_ownership` and `term`. Do we still see the same trend across different levels of `home_ownership` and `term`?

Answer. One possible solution is shown below:

[23]:
```
def get_ct_stra(stra_var):
    ct_stra = pd.crosstab(index=[stra_var,df.verification_status], columns = df.
↪loan_status, margins = True)
    ct_stra["Fully Paid (%)"] = round(ct_stra["Fully Paid"]/
↪ct_stra["All"]*100,2)
    return ct_stra.drop(columns="All").drop("All", level=0)
```

[24]:
```
get_ct_stra(df.home_ownership)
```

[24]:
```
loan_status                         Charged Off  Fully Paid  Fully Paid (%)
home_ownership verification_status
RENT           Not Verified                1072        7133           86.93
               Source Verified              815        4332           84.17
               Verified                     865        4290           83.22
MORTGAGE       Not Verified                 798        6064           88.37
               Source Verified              487        3443           87.61
               Verified                     983        5480           84.79
OWN            Not Verified                 170        1210           87.68
               Source Verified              110         622           84.97
               Verified                     122         611           83.36
OTHER          Not Verified                  10          42           80.77
               Source Verified                1           9           90.00
               Verified                       7          29           80.56
```

[25]:
```
get_ct_stra(df.term)
```

[25]:
```
loan_status                        Charged Off  Fully Paid  Fully Paid (%)
term       verification_status
 36 months Not Verified                   1580       12574           88.84
           Source Verified                 688        6132           89.91
           Verified                        805        6505           88.99
 60 months Not Verified                    470        1875           79.96
           Source Verified                 725        2274           75.83
           Verified                       1172        3905           76.92
```

We can see the trend is preserved when stratified by `home_ownership`. The proportion of fully paid users decreases as the verification status gets less reliable regardless of levels of `home_ownership`. The exception is level `OTHER`. But since we have very few users there, we should not put too much confidence on the results. The trend is no longer preserved in both levels of `term`. We can see the proportion of fully paid users with shorter loan term (36 months) remains the same across different levels of `verification status` at around 88%.

1.2.8 Exercise 6: (5 mts)

Comparing the stratified contigency tables to the unstratified counterparts, which of the following conclusions is correct? Select all that apply.

1. Based on the stratified contigency table, the difference in the probability of full payment between income-verified and not income-verified users might be attributable to the effect of loan term on the probability of full payment.
2. Based on the stratified contigency table, the difference in the probability of full payment between income-verified and not income-verified users is not likely attributable to the effect of home ownership on the probability of full payment.
3. Assume users with no income verified are more likely to borrow short term loans compared to users with verified income and users with short term loans are more likely to pay back their debts. If in fact income verification status has no effect on the probability of paying back one's debt, users with no verified income are on the margin still more likely to pay off their debt.

Answer. 1 is correct. As we see in the stratified table by `term`, the proportion of fully paid loans remains the same across different levels of verification status when `term` is fixed. On the other hand, the stratum-specific probability of full payment is quite different from stratum to stratum. 2 is correct since stratifying the contingency table by `home_ownership` does not significantly change the proportions of fully-paid loans. 3 is correct by definition, and in fact might be the reason why we observed the counter-intuitive trend. We can see the users with no verified income have many more short-term loans, while the stratified contigency table told us that users with longer loan terms are less likely to pay off their debts compared to short-term loan borrowers.

[41]:
```
pd.crosstab(df['term'], df['verification_status'])
```

[41]:
```
verification_status  Not Verified  Source Verified  Verified
term
```

```
36 months                    14154            6820      7310
60 months                     2345            2999      5077
```

1.3 Logistic regression: extending the linear model to binary outcomes (20 mts)

From the chi-square test, we know that verification status is marginally associated with the probability of pay-off. We observed from the contingency table that this probability increases when the information about income becomes less reliable. However, this observation cannot be verified with just the standard chi-square test.

We introduce the logistic regression model to examine if this trend is significant. The logistic regression model is a modified version of the standard linear regression model that deals with classification problems, where the outcomes are binary variables (in this case, whether the loan was paid off or charged off). To understand a logistic regression model, let's first introduce the idea of odds. The odds of an event is the ratio of its success probability to its failure probability. In our case, the odds of fully paying the debt is defined as

$$\text{Odds(Full payment)} = \frac{p}{1-p}.$$

where p is the probability of pay-off.

It turns out that it is easier to work with odds than probabilities. To investigate our question about trend in probability of pay-off p, we assume the odds is a function of income verification status:

$$\log(\text{Odds}) = \beta_0 + \beta_1 * \text{vs}. \tag{1}$$

In order to reflect that there is a trend over different verification statuses, we code different levels of `verification_status` using integers: "Not Verified" as 0, "Source Verified" as 1 and "Verified" as 2; i.e. we are renaming categories of a categorical variable to reflect a clear numerical trend. We assert that this coding scheme implies that if β_1 is positive, then there is an increasing trend of pay-off probability as income verification status becomes more reliable. On the other hand, if β_1 is negative, the opposite is true.

Once we fit the model in (1), we can perform a test to examine if β_1 is significantly different from zero. Note if $\beta_1 = 0$, then p is the same for different levels of `verification_status` and thus no trend is present. If β_1 is significantly different from zero, we can use the estimated sign of β_1 to determine the direction of the trend.

1.3.1 Numerical interpretation of logistic regression (5 mts)

Although the logistic regression model comes with more complexity than standard linear regression, the sign of the regression coefficient still represents the direction of influence of a specific variable: a positive coefficient means the probability of pay-off will increase if the associated variable increases and vice versa.

The other important part of information when interpreting a logistic regression model is the p - value for each regression coefficient. The p-value indicates the result of the following hypothesis test:

$$\boxed{H_0 : \beta = 0 \ vs. \ H_A : \beta \neq 0.}$$

The test determines if the difference in the probability of pay-off is associated with changes in the corresponding variable.

Sometimes you might find that both the p - value and the estimated effect size (regression coefficient) for a variable is small. You should be cautious about this situation as p - values are not the only way to determine if variables are important.

$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \cdot vs$

1.3.2 Geometric interpretation of logistic regression (5 mts)

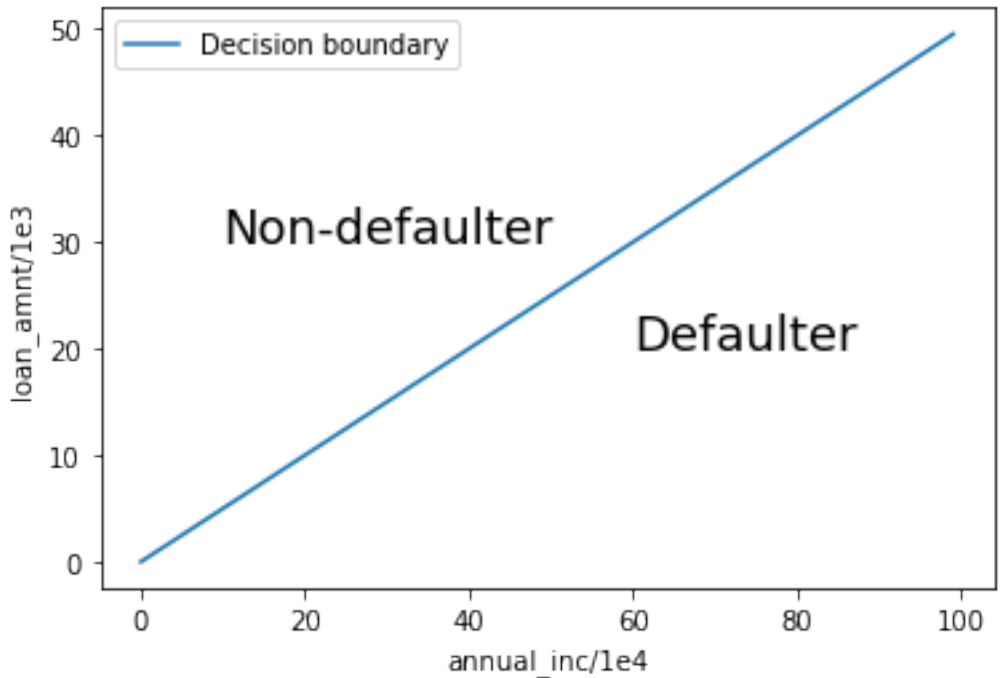
The geometric perspective of logistic regression models is also crucial, as a logistic regression model naturally defines a "decision boundary". If we specify that p is lower than a given decision boundary cut-off, where p_0 (e.g. 0.5) indicates defaulters, then (1) indicates that samples with $\beta_0 + \beta_1 * verification\_status > \log(p_0)/(1-\log(p_0))$ are non-defaulters and $\beta_0 + \beta_1 * verification\_status < \log(p_0)/(1-\log(p_0))$ are defaulters. The line which separates defaulters and non-defaulters, which is defined by the equation $\beta_0 + \beta_1 * verification\_status = \log(p_0)/(1-\log(p_0))$ is the decision boundary determined by model (1) and cut-off value p_0.

Assume we know p is determined by the following logistic model:

$$\log(p/(1-p)) = 2 * loan\_amnt - annual\_inc.$$

Here `annual_inc` is divided by 10^4 and `loan_amnt` is divided by 1000. The line $2 * loan\_amnt = annula\_inc$ in this case separates defaulters from non-defaulters and is the decision boundary with cut-off probability 0.5:

[44]:
```python
x = np.arange(0,100,1)
y = x/2
plt.plot(x, y, label = "Decision boundary");
plt.xlabel("annual_inc/1e4");
plt.ylabel("loan_amnt/1e3");
plt.text(10, 30, "Non-defaulter", fontsize = 18)
plt.text(60, 20, "Defaulter", fontsize = 18)
plt.legend();
```



The shape of the decision boundary in logistic regression is not always a line; oftentimes, it is a curve.

1.3.3 Visualizing the decision boundary (5 mts)

We consider the following logistic model for p:

$$\log(p/(1-p)) = \beta_0 + \beta_1 * loan\_amnt + \beta_2 * annual\_inc + \beta_3 * loan\_amnt^2.$$

We know the corresponding decision boundary with cut-off value $p = 0.5$ is

$$\beta_0 + \beta_1 * loan\_amnt + \beta_2 * annual\_inc + \beta_3 * loan\_amnt^2 = 0.$$

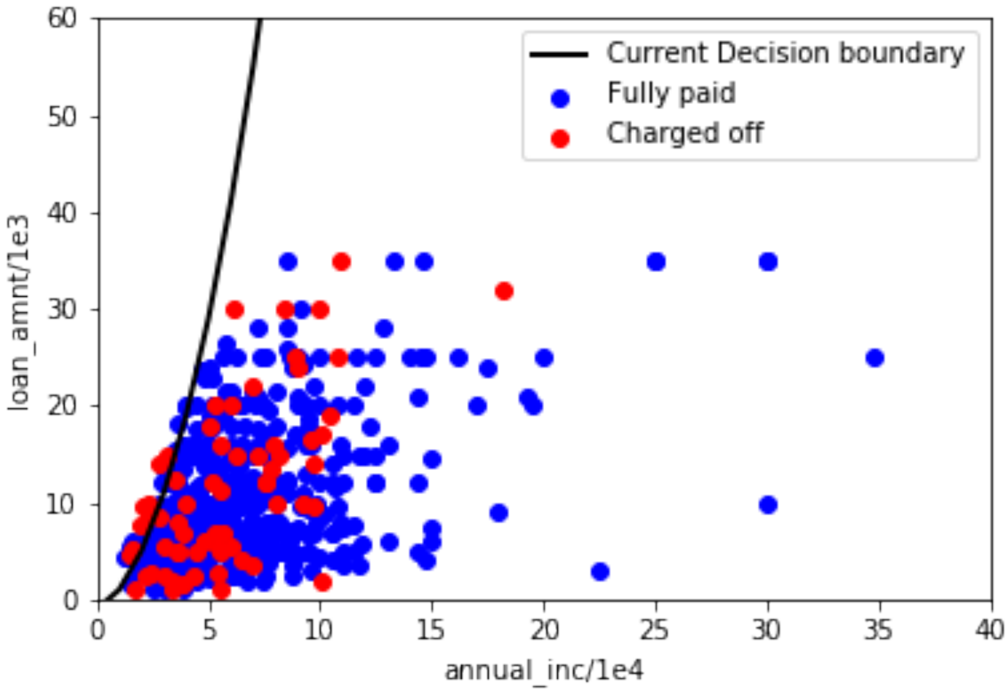
Use the following interactive plot to tweak the parameters and see how the decision boundary changes:

[45]:
```
%matplotlib widget
import ipywidgets as widgets
from ipywidgets import interact

idx_plot = np.random.choice(df.shape[0], size=500, replace = False)
plt_data = df.iloc[idx_plot]

fig, ax = plt.subplots()
```

```
#plt.subplots_adjust(left=0.25, bottom=0.35)
plt.scatter(plt_data[plt_data.loan_status=="Fully Paid"].annual_inc/10000, \
            plt_data[plt_data.loan_status=="Fully Paid"].loan_amnt/1000, \
            c = 'blue', label = 'Fully paid')
plt.scatter(plt_data[plt_data.loan_status=="Charged Off"].annual_inc/10000, \
            plt_data[plt_data.loan_status=="Charged Off"].loan_amnt/1000, \
            c = 'red', label = 'Charged off')
plt.xlim(0,40);
plt.legend();

#plot boundary
beta0 = 1
beta1 = -1
beta2 = 1
beta3 = -1

x_plt = np.arange(0,40,1)
y_plt = (-beta0 - beta1*x_plt - beta3*x_plt**2)/(beta2)
plt.xlabel("annual_inc/1e4")
plt.ylabel("loan_amnt/1e3")
l, = plt.plot(x_plt, y_plt, lw = 2, color="k", label = "Current Decision␣
↪boundary")
plt.ylim(0,60);
plt.legend();

def update(beta0=1, beta1=-1, beta2=1, beta3=-1):
    y_plt = (-beta0 - beta1*x_plt - beta3*x_plt**2)/(beta2)
    l.set_ydata(y_plt);
    fig.canvas.draw_idle();

interact(update, beta0 = widgets.FloatSlider(value=beta0,min=-10,max=10,step=0.
↪5),\
         beta1 = widgets.FloatSlider(value=beta1,min=-10,max=10,step=0.5),
         beta2 = widgets.FloatSlider(value=beta2,min=0.5,max=10,step=0.5),
         beta3 = widgets.FloatSlider(value=beta3,min=-10,max=0,step=0.5));
```

```
Canvas(toolbar=Toolbar(toolitems=[('Home', 'Reset original view', 'home', 'home'), ('Back', 'Ba
```

```
interactive(children=(FloatSlider(value=1.0, description='beta0', max=10.0, min=-10.0, step=0.5
```



1.4 Fitting the logistic regression model (25 mts)

We now fit the logistic model using the `statmodels` package with the specification we described above:

[50]:
```
# code the discrete variable by the specification above
df_log1 = pd.DataFrame(columns=['verification_status','loan_status'])
df_log1['verification_status'] = df.verification_status.cat.codes
df_log1['loan_status'] = df.loan_status.cat.codes
df_log1['Intercept'] = 1

logit = sm.Logit(df_log1['loan_status'],␣
↪df_log1[['Intercept','verification_status']])
logit_res = logit.fit()
print( logit_res.summary() )
```

```
Optimization terminated successfully.
         Current function value: 0.405007
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:            loan_status   No. Observations:                38705
Model:                          Logit   Df Residuals:                    38703
Method:                           MLE   Df Model:                            1
```

```
Date:                    Fri, 29 May 2020   Pseudo R-squ.:                0.002350
Time:                            19:31:55   Log-Likelihood:                -15676.
converged:                           True   LL-Null:                       -15713.
Covariance Type:                nonrobust   LLR p-value:                 8.419e-18
================================================================================
=======
                          coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
-------
Intercept               1.9469      0.022     88.290      0.000       1.904
1.990
verification_status    -0.1460      0.017     -8.600      0.000      -0.179
-0.113
================================================================================
=======
```

This indicates that when `verfication_status` increases by 1, the odds of pay-off decreases by 14%. Since the p - value for the coefficient is smaller than 0.05, we can conclude that the decreasing trend we observed from the table is not likely due to randomness.

An important output above is `Pseudo R-squ.`. This metric is similar to R-squared for linear models. If this number is large, then the variables in the model explain a large portion of the drivers of people's tendency to pay off their debt. In our case, the metric is only 0.2%, which means that we have a lot of room to improve.

1.4.1 Evaluating the numerical coding scheme for `verification_status` (5 mts)

Note that our coding scheme assumes that the difference in the log-odds $p/(1-p)$ between people from any two consecutive levels of `verification_status` is the same. This assumption might be too strong and we ought to check if it is reasonable. We will fit the same model with a different coding scheme for `verification_status`. Instead of mapping levels to integers, we use dummy variables for different levels. With this specification, the difference in log-odds between two consecutive levels is not necessarily a constant. We set "Not Verified" as the reference level. The code and the results are shown below:

[52]:
```
vs_dummy = pd.get_dummies(df.verification_status, drop_first=True)
vs_dummy.head()
```

[52]:

	Source Verified	Verified
0	0	1
1	1	0
2	0	0
3	1	0
4	1	0

[53]:
```
#vs_dummy = pd.get_dummies(df.verification_status, drop_first=True )
df_log1 = pd.concat([df_log1,vs_dummy], axis = 1)
```

```
logit_dummy = sm.Logit(df_log1['loan_status'], df_log1[['Intercept','Source␣
 ↪Verified','Verified']])
logit_dummy_res = logit_dummy.fit()
print( logit_dummy_res.summary() )
```

```
Optimization terminated successfully.
         Current function value: 0.405001
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:            loan_status   No. Observations:                38705
Model:                          Logit   Df Residuals:                    38702
Method:                           MLE   Df Model:                            2
Date:                Fri, 29 May 2020   Pseudo R-squ.:                0.002366
Time:                        19:32:57   Log-Likelihood:                -15676.
converged:                       True   LL-Null:                       -15713.
Covariance Type:            nonrobust   LLR p-value:                 7.140e-17
================================================================================
===
                      coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
---
Intercept           1.9528      0.024     82.741      0.000       1.907
1.999
Source Verified    -0.1696      0.037     -4.558      0.000      -0.242
-0.097
Verified           -0.2916      0.034     -8.566      0.000      -0.358
-0.225
================================================================================
===
```

Based on the results, we can see the difference in log-odds between people from level "Source Verified" and level "Not Verified" is -0.16 while the difference between "Verified" and "Not Verified" is -0.28. The results confirm that the probability of pay-off is indeed decreasing when information of income becomes more reliable. But this also means that the differences in log-odds are not constant and the previous coding scheme might not be reasonable. We can use the `Pseudo R-squ.` as a metric to examine if one coding scheme is better than the other.

1.4.2 Evaluating the logistic regression model (15 mts)

A major drawback of the pseudo R^2 metric is that it cannot be directly translated into the prediction accuracy of a model. Remember that the goal of our whole analysis is to predict whether or not a user is going to pay off his debt. But how do we evaluate if the predication is accurate or not? This question is not as simple as it seems and the reason is because logistic models only produce probabilities while what we observed are binary labels.

A straightforward solution is that you can select a cut-off value for the predicted probability such that people with probabilities smaller than the cut-off value are deemed as defaulters. This converts the probabilities into binary labels, after which we can see if the predicted labels are the same as the observed labels to measure prediction accuracy.

However, this only takes into account a single cut-off value. We ought to summarize the prediction accuracy across all different cut-off values to produce a unified evalution of our model. For logistic models, this is usually implemented via the **receiver operating characteristic (ROC) curve**.

In order to understand a ROC curve, we need to first know the `true positive rate` (TPR) and `true negative rate` (TNR). In our context, TPR measures the proportion of people who paid off their debt that are correctly identified by our model, whereas TNR measures the proportion of defaulters that are identified as defaulters. Each point in the plot indicates the TPR and 1-TNR at a specific cut-off value:

There is a constant trade-off between TPR and TNR. It is usually hard to find a rule with high TPR and TNR at the same time. People will choose to prioritize one of them based on the particular scenario. For instance, in our case, TNR is usually more important than TPR since we would rather avoid lending to defaulters than miss lending to one good candidate.

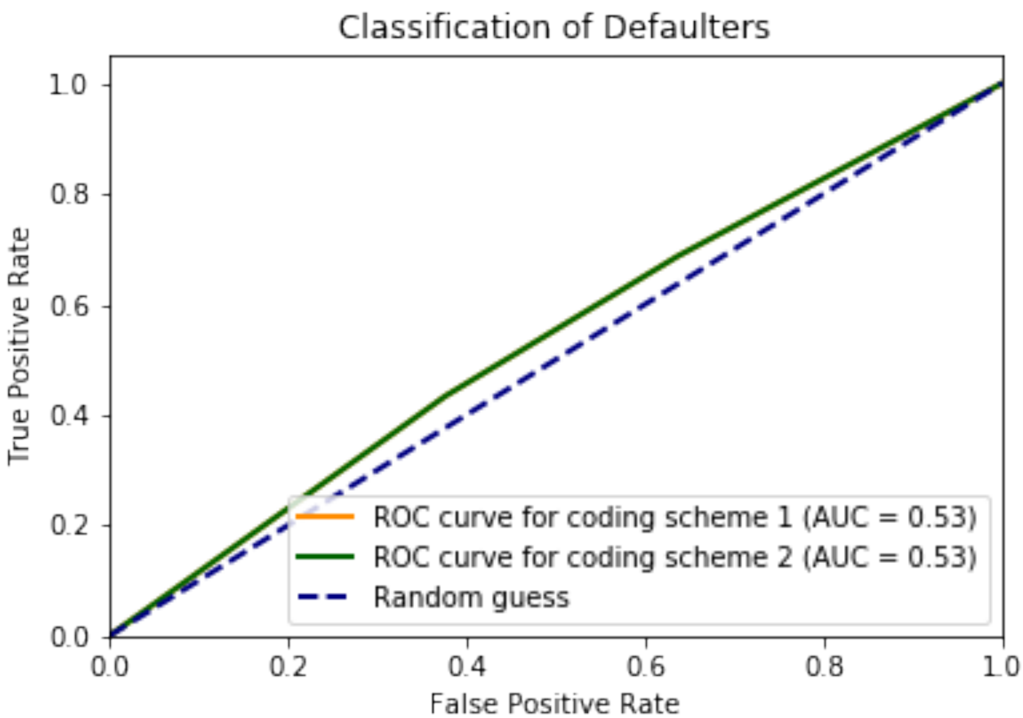
The ROC curve plots TPR against 1-TNR and illustrates the behavior of a logistic model when the cut-off point changes. We can further summarize the whole ROC cruve by the **area under the curve (AUC)**. Roughly speaking, AUC indicates the probability that a randomly selected defaulter will have smaller p than a randomly selected non-defaulter.

If AUC is close to one, the rule is close to a perfect rule. The ROC curve for a random guess rule (set p to 0.5 for all applicants) is the diagonal line in the plot and usually serves as a baseline to check if the logistic model learns any information from the data above and beyond pure randomness. The ROC curves of the above two logistic regression models are shown below:

[54]:
```
predict_p = logit_res.predict(df_log1[['Intercept','verification_status']])
predict_dummy_p = logit_dummy_res.predict(df_log1[['Intercept','Source␣
 ↪Verified','Verified']])
roc_p = roc_curve(df_log1["loan_status"], predict_p)
roc_dummy_p = roc_curve(df_log1["loan_status"], predict_dummy_p)
auc_p = auc( roc_p[0], roc_p[1] )
auc_dummy_p = auc( roc_dummy_p[0], roc_dummy_p[1] )

plt.figure()
lw = 2
plt.plot(roc_p[0], roc_p[1], color='darkorange',
         lw=lw, label='ROC curve for coding scheme 1 (AUC = %0.2f)' % auc_p)
plt.plot(roc_dummy_p[0], roc_dummy_p[1], color='darkgreen',
         lw=lw, label='ROC curve for coding scheme 2 (AUC = %0.2f)' %␣
 ↪auc_dummy_p)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--', label='Random␣
 ↪guess')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
```

```
plt.ylabel('True Positive Rate')
plt.title('Classification of Defaulters')
plt.legend(loc="lower right")
plt.show()
```

```
Canvas(toolbar=Toolbar(toolitems=[('Home', 'Reset original view', 'home', 'home'), ('Back', 'Ba
```



From the figure we find that both ROCs have AUCs quite close to 0.5, confirming that `verification_status` alone is not very predictive of loan status above and beyond random guessing. This also confirms that the first coding scheme has identical performance to the second coding scheme. Since the first coding scheme is more parsimonious (i.e. contains fewer parameters) than the second scheme, we will use the first coding scheme for the rest of this analysis.

1.4.3 Exercise 7: (5 mts)

Based on the results above (pseudo R^2 and AUCs), what can we conclude about the appropriate code scheme for `verification_status`?

Answer. There is no difference in the two metrics when comparing the two different coding schemes. In statistics, we prefer simple models and therefore the first coding scheme is appropriate. Note that in the second coding scheme, we need to estimate one more parameter than in the first coding scheme.

1.5 Adding additional variables to the model (20 mts)

In the previous section, we verified that there is indeed a decreasing trend in pay-off probability associated with more reliable income information. We also performed two model-checking approaches and determined that the integer coding for `verification_status` is adequate. Yet the discovered decreasing trend is counterintuitive. This might be because many other factors are also associated with the probability of pay-off as we have seen in the exploratory analyses. Similar to linear regression, not accounting for these factors will result in completely different answers for the effect of `verification_status`, which is already hinted at in the stratified contingency tables.

We find that except for `annual_inc`, all other variables in the dataset are correlated with `verification_status`. For example, there seems to be a strong correlation between `verification_status` and `loan_amnt`. People with no verified income source tend to borrow less money than people with verified income source. This might be the reason for the decreasing trend associated with `verification_status`, since smaller loans are more likely to be paid off. Let's now incorporate all these variables (including `annual_inc` for now) into the logistic model along with `verification_status` and see if the decreasing trend associated with `verification_status` is still present:

[55]:
```
# Preprocessing the variables
df_log2 = pd.concat([(df.loan_amnt - df.loan_amnt.mean())/df.loan_amnt.std(),\
                    (df.int_rate - df.int_rate.mean())/df.int_rate.std(),\
                    (df.annual_inc - df.annual_inc.mean())/df.annual_inc.
 ↪std(),\
                    pd.get_dummies(df.home_ownership, prefix="home",␣
 ↪drop_first=True), \
                    pd.get_dummies(df.purpose, prefix="purpose",␣
 ↪drop_first=True), \
                    pd.get_dummies(df.grade, prefix="grade",drop_first=True)],␣
 ↪axis=1)
df_log2["verification_status"] = df.verification_status.cat.codes
df_log2["emp_length"] = df.emp_length.cat.codes
df_log2["term"] = df.term.cat.codes
df_log2["Intercept"] = 1
```

[56]:
```
logit_full1 = sm.Logit(df.loan_status.cat.codes, df_log2)
logit_full1_res = logit_full1.fit()
print(logit_full1_res.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.378075
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                38705
Model:                          Logit   Df Residuals:                    38676
Method:                           MLE   Df Model:                           28
Date:                Fri, 29 May 2020   Pseudo R-squ.:                 0.06869
```

```
Time:                         19:36:42   Log-Likelihood:                -14633.
converged:                        True   LL-Null:                       -15713.
Covariance Type:             nonrobust   LLR p-value:                     0.000
================================================================================
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
loan_amnt                     -0.0075      0.019     -0.389      0.697
-0.045       0.030
int_rate                      -0.4050      0.054     -7.454      0.000
-0.511      -0.299
annual_inc                     0.3615      0.031     11.641      0.000
0.301       0.422
home_MORTGAGE                  0.0382      0.035      1.095      0.273
-0.030       0.107
home_OWN                       0.0209      0.060      0.348      0.728
-0.097       0.138
home_OTHER                    -0.4454      0.268     -1.660      0.097
-0.971       0.080
purpose_credit_card            0.0364      0.100      0.363      0.717
-0.160       0.233
purpose_debt_consolidation    -0.2190      0.091     -2.405      0.016
-0.398      -0.041
purpose_educational           -0.5663      0.178     -3.186      0.001
-0.915      -0.218
purpose_home_improvement      -0.1624      0.107     -1.523      0.128
-0.371       0.047
purpose_house                 -0.3556      0.171     -2.077      0.038
-0.691      -0.020
purpose_major_purchase        -0.0024      0.114     -0.021      0.983
-0.227       0.222
purpose_medical               -0.3945      0.141     -2.794      0.005
-0.671      -0.118
purpose_moving                -0.4147      0.150     -2.766      0.006
-0.708      -0.121
purpose_other                 -0.4050      0.099     -4.105      0.000
-0.598      -0.212
purpose_renewable_energy      -0.7242      0.286     -2.530      0.011
-1.285      -0.163
purpose_small_business        -0.9414      0.105     -8.961      0.000
-1.147      -0.735
purpose_vacation              -0.4371      0.179     -2.439      0.015
-0.788      -0.086
purpose_wedding                0.1348      0.142      0.948      0.343
-0.144       0.414
grade_B                       -0.2733      0.074     -3.709      0.000
```

```
-0.418      -0.129
grade_C                          -0.3549      0.103     -3.448      0.001
-0.557      -0.153
grade_D                          -0.3775      0.132     -2.860      0.004
-0.636      -0.119
grade_E                          -0.2973      0.160     -1.859      0.063
-0.611       0.016
grade_F                          -0.3238      0.194     -1.669      0.095
-0.704       0.057
grade_G                          -0.1802      0.238     -0.757      0.449
-0.647       0.286
verification_status               0.0082      0.020      0.411      0.681
-0.031       0.047
emp_length                       -0.0147      0.004     -3.301      0.001
-0.023      -0.006
term                             -0.4520      0.037    -12.081      0.000
-0.525      -0.379
Intercept                         2.6163      0.123     21.216      0.000
2.375       2.858
==============================================================================
==============
```

1.5.1 Exercise 8: (5 mts)

Based on the results of the logistic model with all variables included, which of the following conclusions is correct? Select all that apply.

1. There is no evidence that supports the association between probability of pay-off and verification status after accounting for other variables.
2. Loans borrowed for small business have the smallest pay-off probability.
3. The odds of pay-off in users with LendingClub grade B decreases by 24% compared to users with grade A after accounting for other variables.

Answer. 1 is correct as the p - value for the variable `verification_status` is not smaller than 0.05. 2 is not correct as the statement does not account for the effects of other variables. 3 is correct as we can calculate it using the estimated coefficient.

1.5.2 Why do we observe a marginal trend associated with verification_status? (10 mts)

Interestingly, we find that the regression coefficient for `verification_status` is now positive and the p - value for it is very large (0.68). This indicates that after adjusting for all other variables in the dataset, `verification_status` is no longer significantly associated with the probability of pay-off. Let's now find out what confounding variables were responsible for introducing the initial decreasing trend that we observed.

We can investigate this problem using the regression results along with the correlation between `verification_status` and all other variables. First let's take a look at the pairwise relationship between `verification_status` and all other variables:

[57]:
```
def get_rctable(var1, var2):
    res = pd.crosstab(df[var2], df[var1])
    chi2, p, dof, ex = chi2_contingency(res)
    output = round( res.div(res.sum(axis = 1),axis = 0)*100, 2 )
    return output.style.set_caption("%s vs. %s: Chi-square p-value=%.2f" %␣
 ↪(var1,var2,p))
```

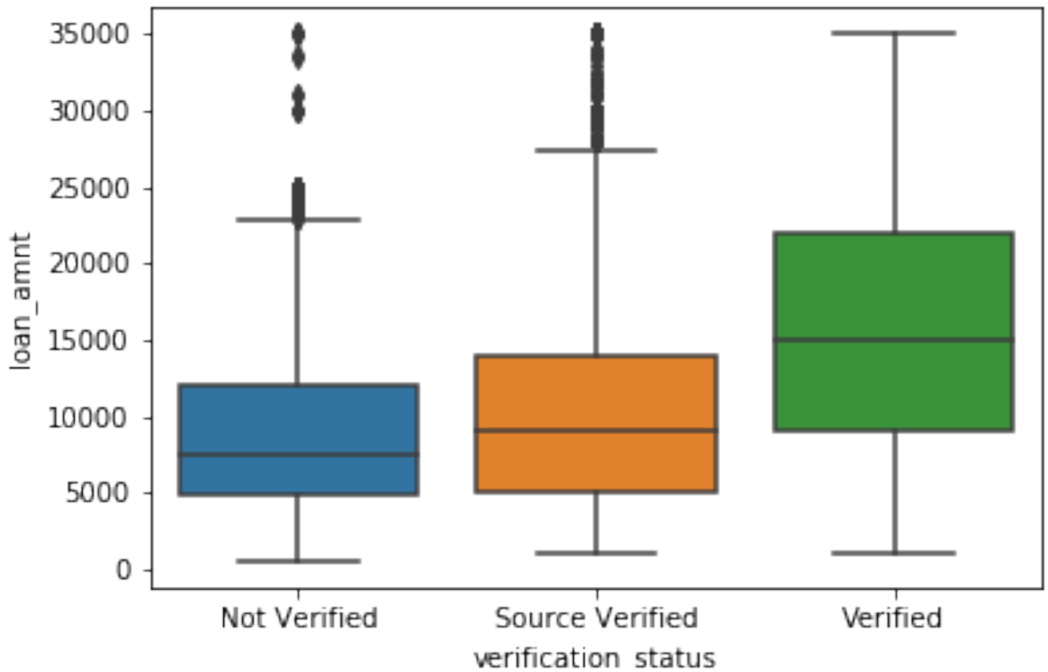
[58]:
```
display(get_rctable("emp_length", "verification_status"))
display(get_rctable("home_ownership", "verification_status"))
display(get_rctable("purpose", "verification_status"))
display(get_rctable("term", "verification_status"))
display(get_rctable("grade", "verification_status"))
```

```
<pandas.io.formats.style.Styler at 0x7fb998cd4da0>

<pandas.io.formats.style.Styler at 0x7fb99e23a5f8>

<pandas.io.formats.style.Styler at 0x7fb9d96f7898>

<pandas.io.formats.style.Styler at 0x7fb99b74fc50>

<pandas.io.formats.style.Styler at 0x7fb9af5e12e8>
```

[59]:
```
sns.boxplot(y="loan_amnt", x='verification_status', data = df)
```

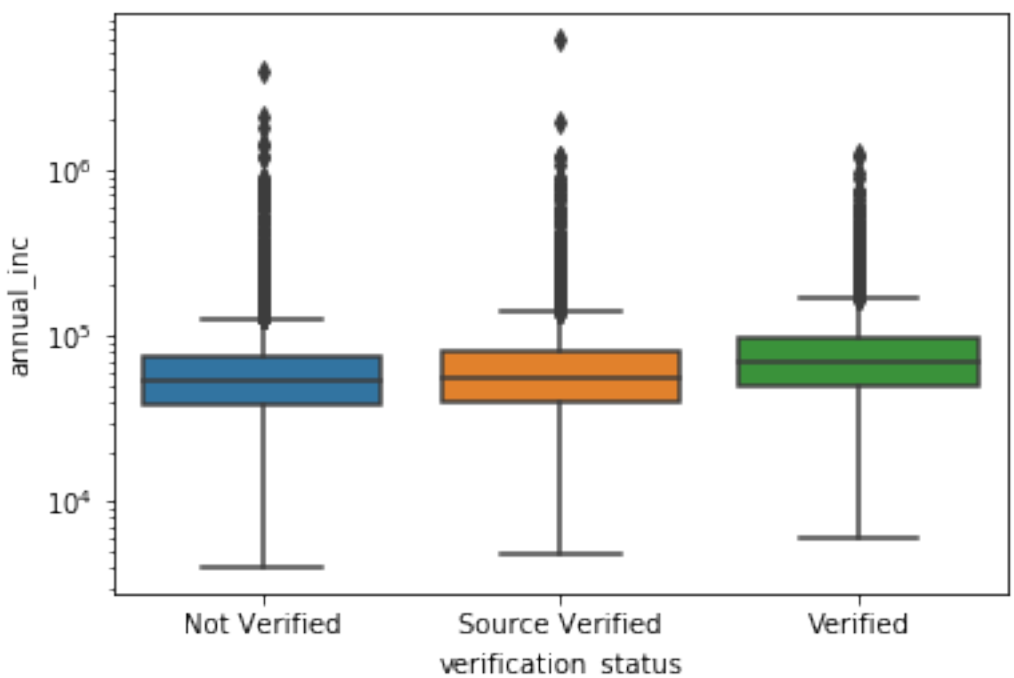
```
Canvas(toolbar=Toolbar(toolitems=[('Home', 'Reset original view', 'home', 'home'), ('Back', 'Ba
```

[59]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb99b5620b8>
```



[60]:
```
sns.boxplot(y="annual_inc", x='verification_status', data = df).
↪set_yscale('log')
```

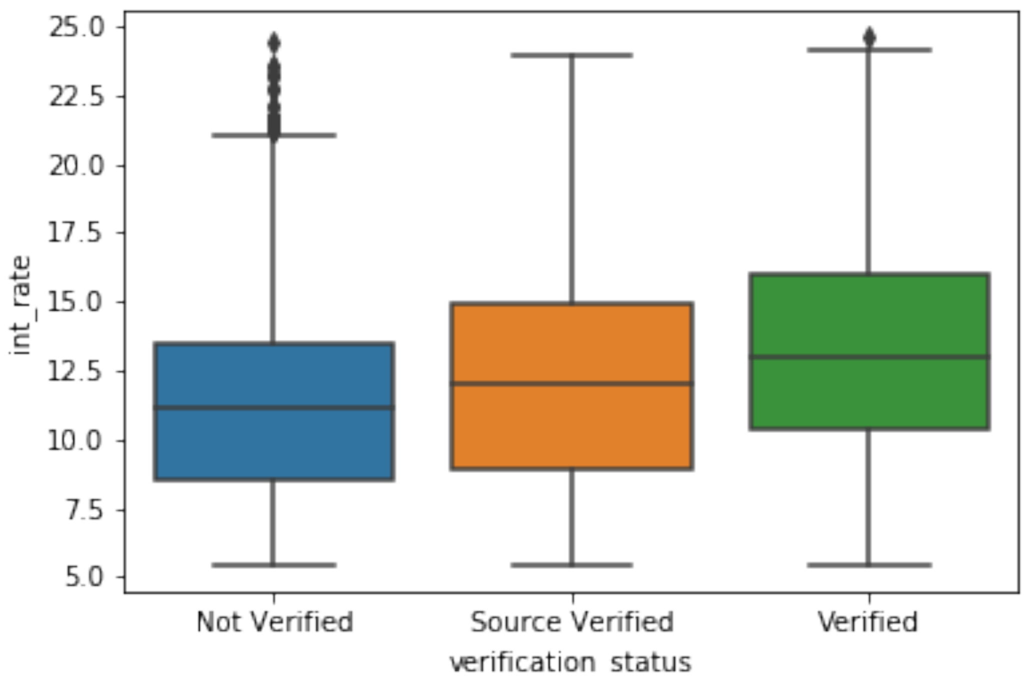
```
Canvas(toolbar=Toolbar(toolitems=[('Home', 'Reset original view', 'home', 'home'), ('Back', 'Ba
```



[61]:
```python
sns.boxplot(y="int_rate", x='verification_status', data = df)
```

```
Canvas(toolbar=Toolbar(toolitems=[('Home', 'Reset original view', 'home', 'home'), ('Back', 'Ba
```

[61]: <matplotlib.axes._subplots.AxesSubplot at 0x7fb998e0cd68>



1.5.3 Exercise 9: (5 mts)

Based on the relationships we observed in the contingency tables and the boxplots above, which of the following statements are correct? Select all that apply.

A. The marginal trend is that people with verified income tend to borrow loans with higher interest rates

B. The marginal trend is that people with verified income tend to have longer employment length

C. The marginal trend is that people with verified income tend to have mortgages

Answer. A and B are both correct since high interest rate and long employment length are both associated with lower probability of pay-off. C is not correct since home ownership is not significantly associated with probability of pay-off after controlling for other variables.

1.6 Do we need to include `verification_status`? (10 mts)

From the above, we might reach the conclusion that `verification_status` does not matter when we want to predict if a borrower is going to default on his/her debt. But does this mean that in the future, we can just ignore the verification status of an applicant when he/she is filing for a loan application through LC?

1.6.1 Exercise 10: (5 mts)

Which of following statements is true regarding what we should do with `verification_status`, based on the results we have so far?

A. Since the p - value for `verification_status` is large, we can remove it from our classification model and in the future, we do not need to collect this part of information from applicants.

B. Although the p - value for `verification_status` is large, the estimated regression coefficient for it is non-zero. We should keep the variable in the model and continue to collect the information in the future.

C. We should carry out an additional evaluation to compare the prediction accuracies of the model with and without `verification_status` before we can make any further decision.

D. None of the above.

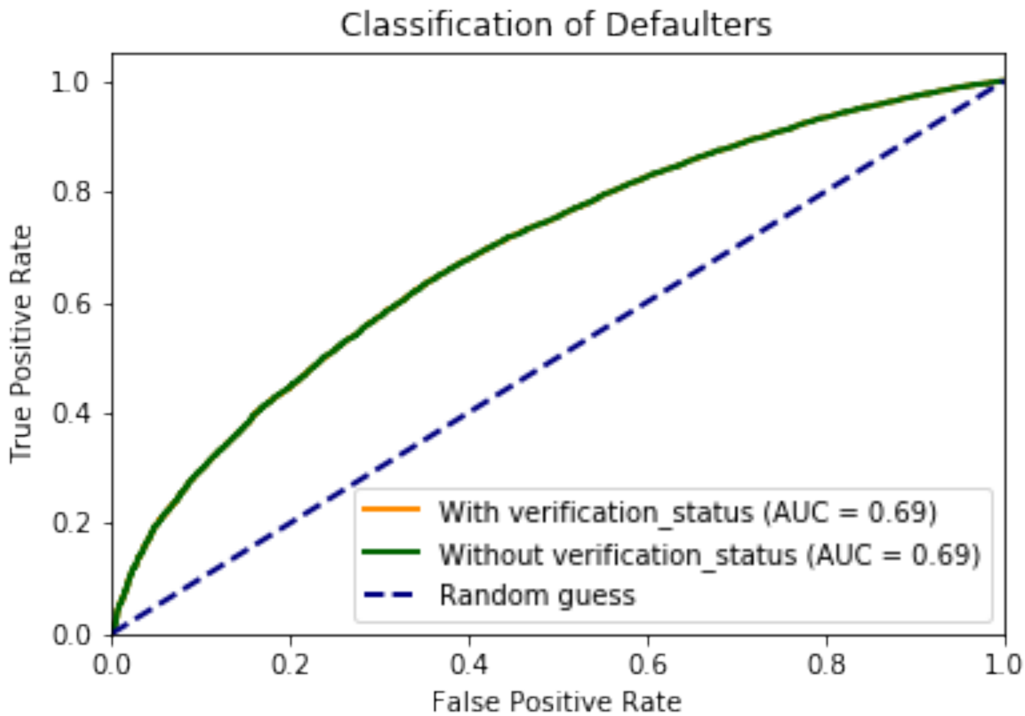
Answer. A and B are both incorrect. We should not only base our decision based on the p - value alone. C is correct.

Let's evalute the prediction accuracy using ROC. Theresults are shown below:

[62]:
```
predict_withvs = logit_full1_res.predict(df_log2)
logit_full_novs = sm.Logit(df.loan_status.cat.codes, df_log2.loc[:, df_log2.
 ↪columns != 'verification_status'])
predict_novs = logit_full_novs.fit(disp=0).predict(df_log2.loc[:, df_log2.
 ↪columns != 'verification_status'])

roc_vs = roc_curve(df.loan_status.cat.codes, predict_withvs)
roc_novs = roc_curve(df.loan_status.cat.codes, predict_novs)
auc_vs = auc( roc_vs[0], roc_vs[1] )
auc_novs = auc( roc_novs[0], roc_novs[1])

plt.figure()
lw = 2
plt.plot(roc_vs[0], roc_vs[1], color='darkorange',
         lw=lw, label='With verification_status (AUC = %0.2f)' % auc_vs)
plt.plot(roc_novs[0], roc_novs[1], color='darkgreen',
         lw=lw, label='Without verification_status (AUC = %0.2f)' % auc_novs)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--', label='Random␣
 ↪guess')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
```

```
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Classification of Defaulters')
plt.legend(loc="lower right")
plt.show()
```

```
Canvas(toolbar=Toolbar(toolitems=[('Home', 'Reset original view', 'home', 'home'), ('Back', 'Ba
```



1.6.2 Exercise 11: (5 mts)

Based on all the results above, what can we conclude about the verification process?

Answer. We find that the AUC for the model with `verification_status` included is the same as the model without `verification_status` included (0.69). Since we always want our model to be as simple as possible, this means that we should not include `verification_status` as a variable when predicting if a user is likely to pay off his debt.

1.7 Conclusions (5 mts)

In this case, we investigated whether the verification status of applicants' income source is important when predicting if an applicant will pay off his/her debt. We constructed a logistic model to first examine the marginal relationship between verification status and probability of pay-off.

We discovered a counterintuitive trend: less reliable income source information was correlated with higher pay-off probability. However, after fitting a larger logistic model with all independent variables that were available, we found that this trend was introduced via confounding effects of interest rate, annual income, and loan term.

After accounting for these variables, verification status was no longer significantly associated with pay-off probability. Based on p - values and the model ROC, we concluded that the verification process is irrelevant and we could potentially remove it from the required items and simplify the lending process.

1.8 Takeaways (5 mts)

Logistic regression naturally extends the concept of linear regression and is the benchmark model for performing classification tasks. The coefficients have similar interpretation as that of linear regression. ROC curves are then useful for comparing different classification models.

[ ]:
```

```