* See the distribution of gender in our data

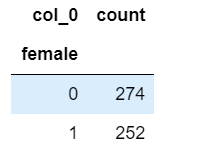
# load wage1 data from wooldridge package

wage1 = woo.dataWoo('wage1')

# Make a crosstab

**pd.crosstab**(index=wage1['female'],

columns="count") # Name the count column



* We are interested in the wage differences by gender. The regression equation will be the following formula

1618542141(1)

m1 = smf.ols(formula='np.log(wage) ~ female + educ + exper + tenure', data=wage1)

m1 = m1.fit()

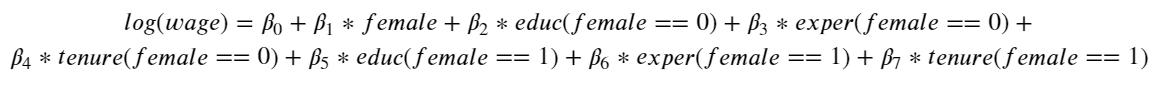
#You can also filter your data and create two separate equations but the most efficient way is to add the subset option inside the lm command data=subset()

m2 = smf.ols(formula='np.log(wage) ~ educ + exper + tenure',

data=wage1, **subset=(wage1['female'] == 0)**).fit()

m3 = smf.ols(formula='np.log(wage) ~ educ + exper + tenure',

data=wage1, **subset=(wage1['female'] == 1)**).fit()



m4 = smf.ols(formula=' np.log(wage) ~ educ\*female + exper\*female + tenure\*female ', data=wage1).fit()

# Put these models in stargazer table with the intercept at the bottom see the table

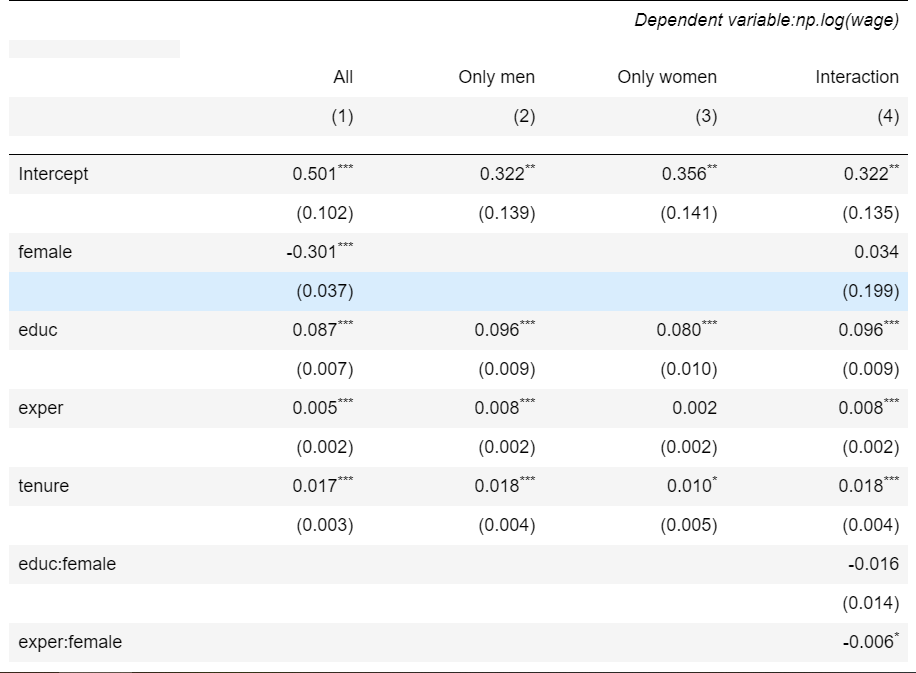
models = Stargazer([m1, m2, m3, m4])

models.title('Regression on Wages')

models.custom\_columns(['All', 'Only men', 'Only women', 'Interaction'], [1, 1, 1, 1])

models.covariate\_order(['Intercept', 'female' , 'educ' , 'exper', 'tenure', 'educ:female', 'exper:female','tenure:female'])

HTML(models.render\_html())



* Dummy variables and arithmetic formulas into a regression

1618543153(1)

reg = smf.ols(formula='np.log(wage) ~ married\*female + educ + exper + I(exper\*\*2) + tenure +I(tenure\*\*2)', data=wage1)

results = reg.fit()

#reg1 and reg2 are the same

reg1 = smf.ols(formula='np.log(wage) ~ married + **educ + female** + I(educ\*female) + exper +I(exper\*\*2) + tenure +I(tenure\*\*2)', data=wage1)

results1 = reg1.fit()

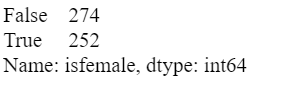
reg2 = smf.ols(formula='np.log(wage) ~ married + **educ\*female** + exper + I(exper\*\*2) + tenure +I(tenure\*\*2)', data=wage1)

results2 = reg2.fit()

* Boolean Variables

wage1['isfemale'] = (wage1['female'] == 1)

wage1['isfemale'].value\_counts()



# regression with boolean variable:

m6 = smf.ols(formula='np.log(wage) ~ isfemale+ educ + exper + tenure', data=wage1)

m6 = m6.fit()

m6s = Stargazer([m6])

m6s.covariate\_order(['Intercept','isfemale[T.True]' , 'educ' , 'exper', 'tenure'])

m6s.rename\_covariates({'isfemale[T.True]': 'Female:True'})

HTML(m6s.render\_html())

* Categorical Variables

CPS1985 = pd.read\_csv('C:/Users/DELL/Desktop/CPS1985.csv')

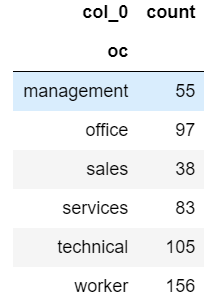
# rename variable to make outputs more compact:

CPS1985['oc'] = CPS1985['occupation']

#*Occupation has 5 categories*

freq\_occupation = pd.crosstab(CPS1985['oc'], columns='count')

freq\_occupation



* *You can easily transform any variable into a categorical variable using the function* ***C()*** *in the definition of the formula*

# directly using categorical variables in regression formula:

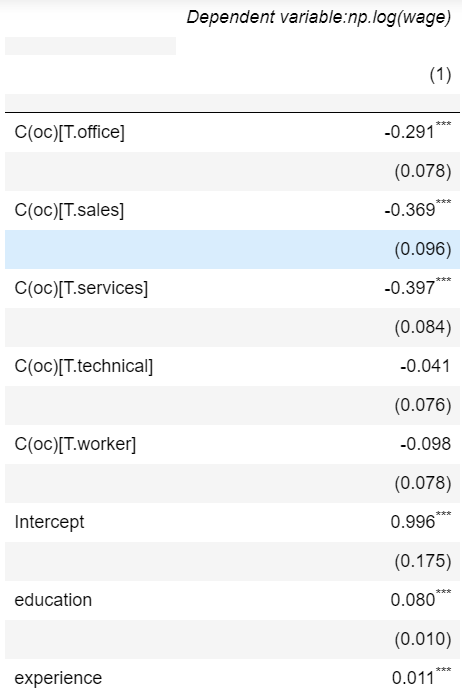
m7 = smf.ols(formula='np.log(wage) ~ education + experience + **C(oc)**', data=CPS1985)

m7 = m7.fit()

# print regression table:

m7s = Stargazer([m7])

HTML(m7s.render\_html())



* When you use categorical variables that have many categories, you have to choose a *reference category* and this is the ommitted variable that you use to avoid colinearity. By default the first category is left out in Python but we can use a second argument in the C() command where we provide a new reference group somegroup with the using the command **Treament**("somegroup").

#Changing a new reference category

# rerun regression with different reference category:

reg\_newref = smf.ols(formula='np.log(wage) ~ education + experience + '

'**C**(gender, **Treatment**("male")) + '

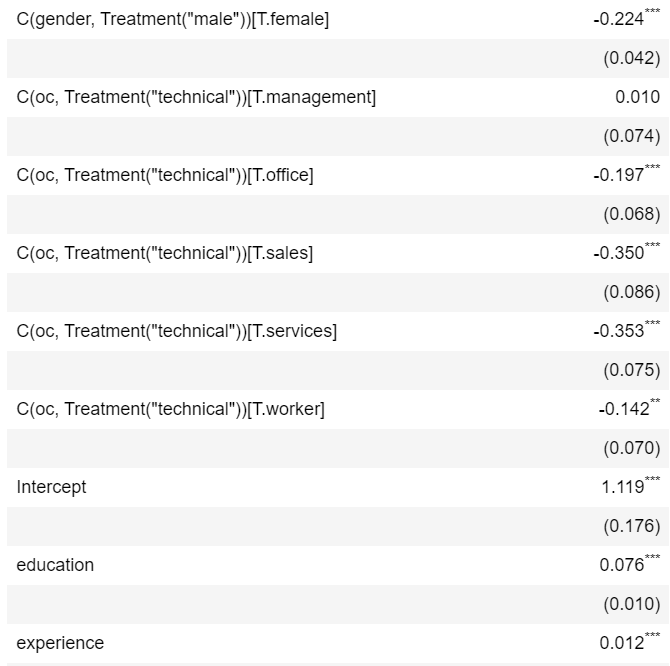
'**C**(oc,**Treatment**("technical"))', data=CPS1985)

m8 = reg\_newref.fit()

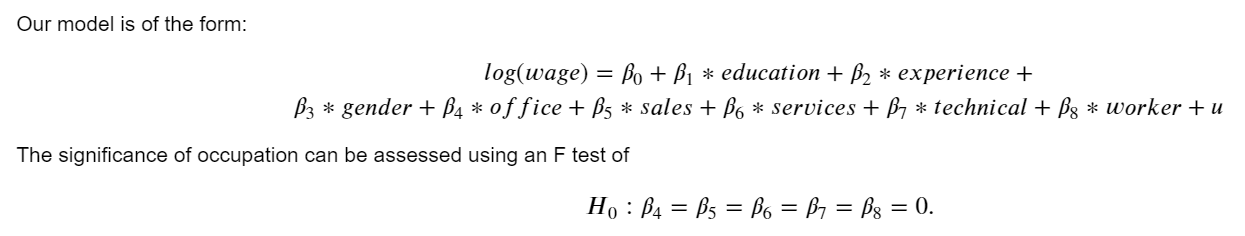
# print regression table:

m8s = Stargazer([m8])

HTML(m8s.render\_html())



* **Anova tables**



# run regression:

reg = smf.ols(

formula='np.log(wage) ~ education + experience + gender + occupation',

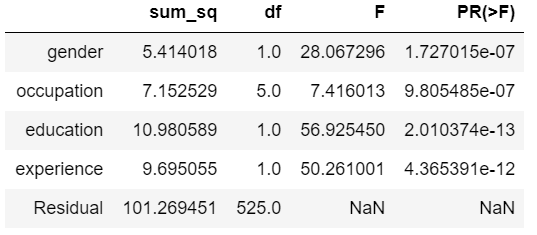
data=CPS1985)

*results* = reg.fit()

# ANOVA table:

table\_anova = sm.stats.anova\_lm(*result*s, typ=2)

table\_anova



# **Numeric variables into categories**

Sometimes we need to make numerical variables into categories because a linear relation with the dependent variable seems implausible or the interpretation is inconvenient. Or we simply want to have a different interpretation.

lawsch85 = woo.dataWoo('lawsch85')

# define *cut points* for the rank:

***cutpts*** = [0, 10, 25, 40, 60, 100, 175]

# create categorical variable containing ranges for the rank:

lawsch85['rc'] = **pd.cut**(lawsch85['rank'], **bins= *cutpts***,

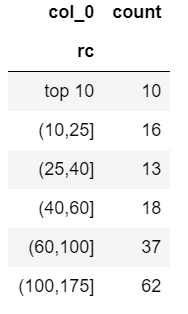
**labels**=['top 10', '(10,25]', '(25,40]',

'(40,60]', '(60,100]', '(100,175]'])

# display frequencies:

freq = pd.crosstab(lawsch85['rc'], columns='count')

freq



# run regression:

reg = smf.ols(formula='np.log(salary) ~ C(rc, Treatment("(100,175]")) +'

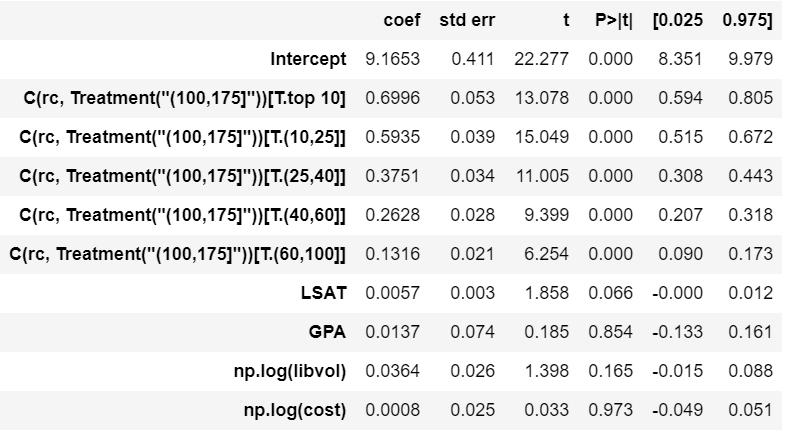
'LSAT + GPA + np.log(libvol)+ np.log(cost)',

data=lawsch85)

results = reg.fit()

# print regression table

results.summary()



* **Categorical dependent variables**

*# logit model:*

reg\_logit = **smf.logit**(formula='y ~ x1 + x2 + ...+ xn',

data=mydata)

results\_logit = reg\_logit.fit(**disp=0**)

# *Estimate probit model*

reg\_probit = **smf.probit**(formula='y ~ x1 + x2 + ...+ xn',

data=mydata)

results\_probit = reg\_probit.fit(**disp=0**)

* Create interaction variables: Option 1 and 2 are the same

# - Option 1

# As seen in class you can use the \* function to create interactions of your dummy variables,

# when using \* you do not need to add the variables alone Phyton does that automatically for you.

m7 = smf.ols(formula = 'np.log(wage) ~ educ + exper + tenure + **married \* black** + south + urban', data = wage2).fit()

# - Option 2

# The other way is using colon **:** for the interaction,

# in which case you do need to add the main effects or main dummy variables, in this example, married and black.

m8 = smf.ols(formula = 'np.log(wage) ~ educ + exper + tenure + **married** + **black** + south + urban + **married:black**', data = wage2).fit()

# - Option 3

# create the interaction factor variable before the regression, this creates four categories, married\_black(00, 10, 01, 11)

**wage2['marital\_race'] = wage2['married'].astype(int).astype('str')+'\_'+wage2['black'].astype(int).astype('str')**

m9 = smf.ols(formula = 'np.log(wage) ~ educ + exper + tenure + south + urban + marital\_race', data = wage2).fit()

#view the results using a stargazer table

st=Stargazer([m7, m8, m9])

st.covariate\_order(['educ' , 'exper', 'tenure', 'south', 'urban',

'married' , 'black', 'married:black', 'marital\_race[T.1\_0]',

'marital\_race[T.0\_1]','marital\_race[T.1\_1]','Intercept'])

from IPython.core.display import HTML

HTML(st.render\_html())