Saint Petersburg State University of Information Technologies,

Mechanics and Optics

Laboratory work report 1

Confidence interval for probabilities of discrete choice

on course: Discrete decision making

(course name)

Work completed by: Bizhumanova А.

(last name, first name)

academic group C4215c

(group number)

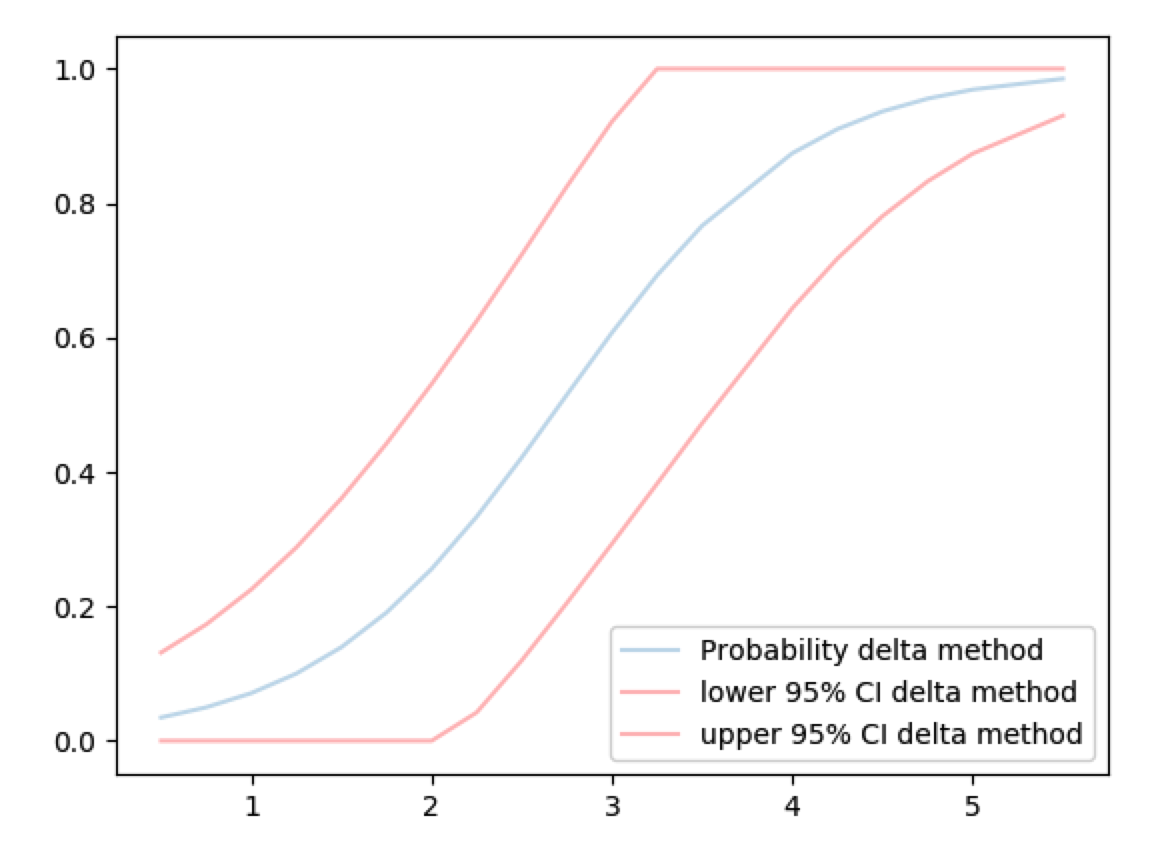
Saint-Petersburg

2018

## Part 1:

Find confidence interval for probabilities of passing exam for 1-5 Hours of studying using Delta Method and regularized logit fit function.

x = [0.50, 0.75, 1.00, 1.25, 1.50, 1.75, 1.75, 2.00, 2.25, 2.50, 2.75, 3.00, 3.25, 3.50, 4.00, 4.25, 4.50, 4.75, 5.00, 5.50]  
y = [0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1]

X = sm.add\_constant(x)  
  
logit = sm.Logit(y,X).fit\_regularized(disp=**False**)  
proba = (logit.predict(X))  
  
*# estimate confidence interval for predicted probabilities*cov = logit.cov\_params()  
gradient = (proba \* (1 - proba) \* X.T).T *# matrix of gradients for each observation*std\_errors = np.array([np.sqrt(np.dot(np.dot(g, cov), g)) **for** g **in** gradient])  
  
c = 1.96 *# multiplier for confidence interval*upper = np.maximum(0, np.minimum(1, proba + std\_errors \* c))  
lower = np.maximum(0, np.minimum(1, proba - std\_errors \* c))  
  


## Part 2:

1.Generate large sample from Exam task using uniform distribution generator and logit regression

2.Generate N samples of size len(hours) from main distribution

3.Estimate confidence interval for probabilities of passing exam for 1-5 Hours of studying  using Bootstrapping

4.Compare results (confidence interval) with Task 1

rnd\_hrs = np.random.uniform(0,5,1000)  
rnd\_hrs = np.array(rnd\_hrs)  
  
*# rnd\_X = sm.add\_constant(rnd\_hrs)  
# rnd\_proba = (logit.predict(rnd\_X))*preds = []  
  
**for** i **in** range(1000):  
 **try**:  
  
 pred = logit.predict(X)  
 new\_y = [1 **if** np.random.random() < p **else** 0 **for** p **in** pred]  
 new\_logit = sm.Logit(new\_y, X).fit\_regularized(disp=**False**)  
 proba\_new = new\_logit.predict(X)  
 *# print(proba\_new)* preds.append(proba\_new)  
  
 **except**:  
**pass**print(len(preds))  
df\_preds = pd.DataFrame(preds)  
*# print(df\_preds)*q\_0\_5 = df\_preds.quantile(q=0.075, axis=0)  
q\_0\_95 = df\_preds.quantile(q=0.935, axis=0)

