به نام خدا

علی فتحی ۹۴۱۰۹۲۰۵ تمرین کامپیوتری سری اول درس استنتاج علّی

سوال اول، بخش اول:

(آ) یک مدل SCM دو متغیری را در نظر بگیرید که در آن $Y = X + bX^3 + sign(N)|N|^q$ که X و N متغیرهای تصادفی مستقل از توزیع نرمال باشند و پارامترهای b و q به ترتیب کنترل کننده‌ی خطی بودن و گاوسی بودن مدل هستند. با استفاده از مدل ارائه شده در بالا، نمودار توزیع های $p(x,y), p(x|y), p(y|x)$ برای دو مدل خطی و غیر خطی با نویز گاوسی را رسم کنید.

کد نوشته شده با نام SCMCausal.py آورده شده است (کد تمامی بخش‌های این تمرین در غالب‌ها توابعی در همین کد آورده شده‌اند). تابع موبوط استفاده شده در این بخش، stochasticPlot نام دارد و به صورت زیر است:

```
# Function for Plotting
def stochasticPlot(x, y, kind):
    # Functionality:
    # kind = 0: plots p(x,y) on (x,y)
    # kind = 1: plots p(x) on x
    # kind = 2: plots p(y) on y
    # kind = 3: plots p(y|x) on (x,y)
    # kind = 4: plots p(x|y) on (x,y)
    # kind = 5: plots p(x,y - E(Y|x)) on (x,y)
    # kind = 6: plots p(x - E(X|y),y) on (x,y)
    # kind = 7: plots p(y|x - E(Y|x)) on (x,y)
    # kind = 8: plots p(x|y - E(X|y)) on (x,y)
```

که کاربری‌های فوق را داراست.

نحوه‌کار تابع برای رسم بدین صورت است که تمام داده‌ها را به صورت یک histogram2D درمی‌آورد که در اینجا تعداد bin ها در هر راستا برابر ۵۰۰ عدد است (و در واقع فضای داده‌ها به ۲۵۰۰۰۰ قسمت تقسیم شده است).
(داده‌ها با نمایش نقطه‌ای به شکل (x,y) رسم نشده اند و بصورت رنگ‌ها در فضا در می‌آیند)

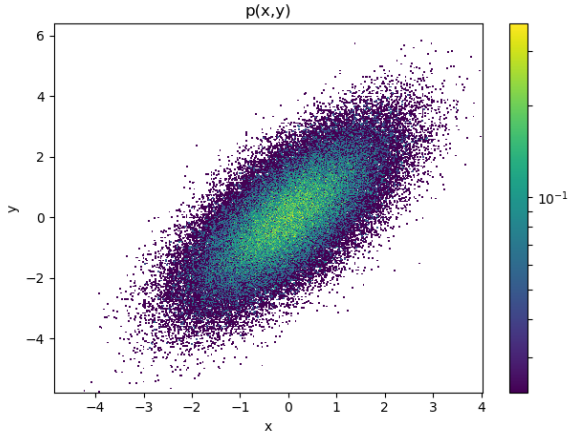
برای اجرای این تابع، تابع اجرای بخش اول سوال صدا می‌شود که به شکل زیر است:

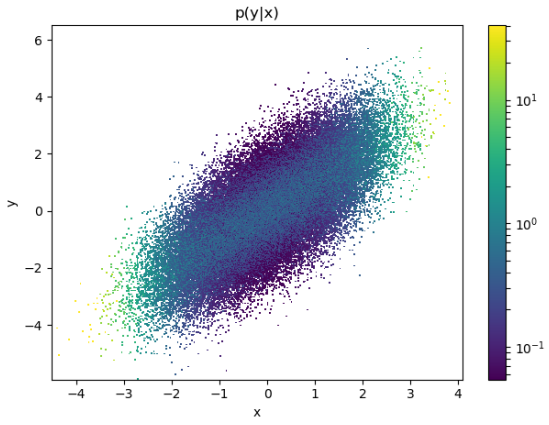
```
# Part 1 Session 0, Plotting
def part1Plotting():
    # Generating X Distribution: Nx = N(0, 1)
    meanNX = 0
    sigmaNX = 1
    NX = np.random.normal(meanNX, sigmaNX, 100000)
    # Calculating Y = X + bX^3 + sign(X)*|X|^q
    X = NX
    Y = yGen(X, b=0, q=2)
    stochasticPlot(X, Y, kind=0)
```

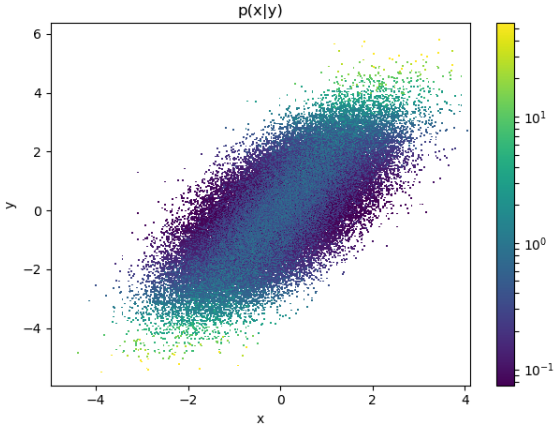
تعداد داده‌ها برای رسم دقیق‌تر اشکال، برابر با یک مقدار بزرگ، برای مثال ۱۰۰۰۰۰ قرار داده شده اند.

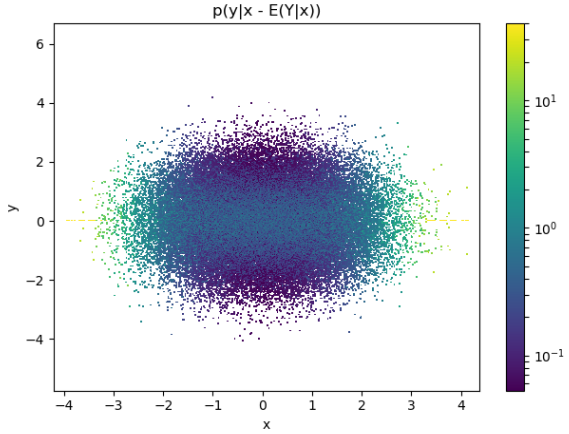
برای این سوال تنها kind را برابر ۰، ۳ و ۴ قرار می‌دهیم و به عنوان اضافات، مقادیر ۷ و ۸ را هم می‌اوریم. نتایج برای سه مدل داده‌سازی به صورت زیر اند:

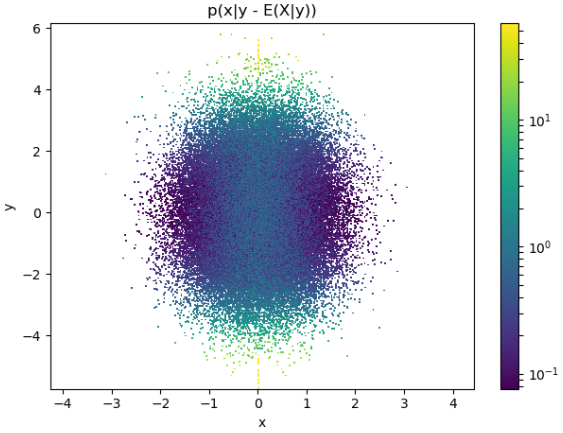
الف) داده‌های خطی با نویز گاوسی، $b = 0$ و $q = 1$، $Y = X + \mathcal{N}_y$



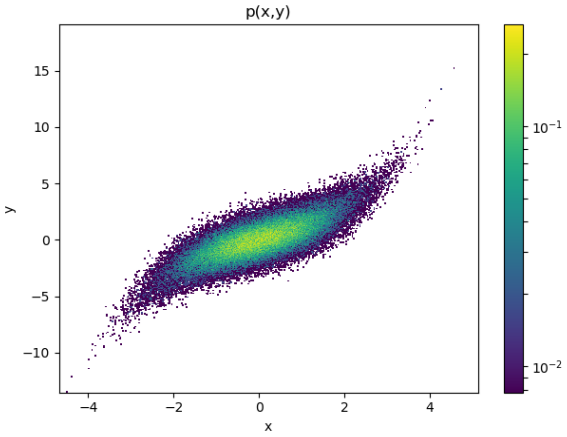


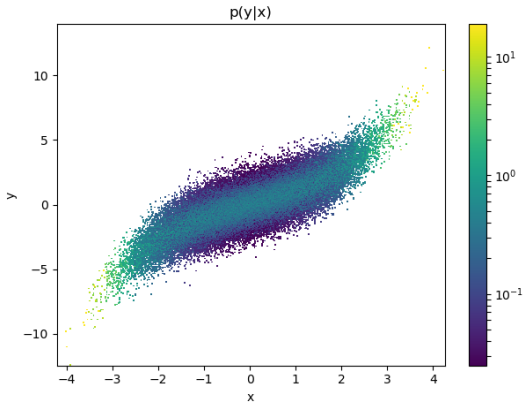


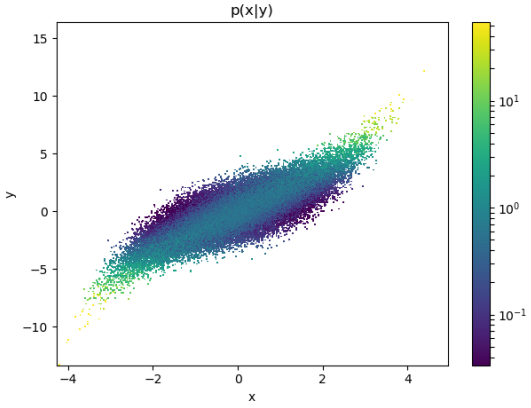


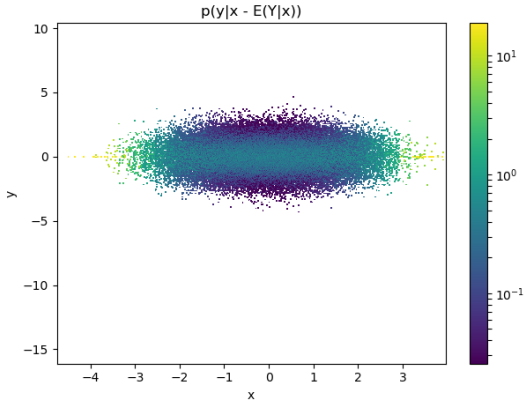


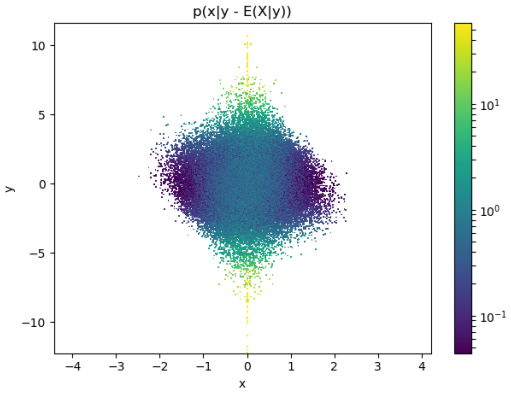
ب) داده‌های غیرخطی با نویز گاوسی، $b = 0.1$ و $q = 1$، $Y = X + bX^3 + \mathcal{N}_y$



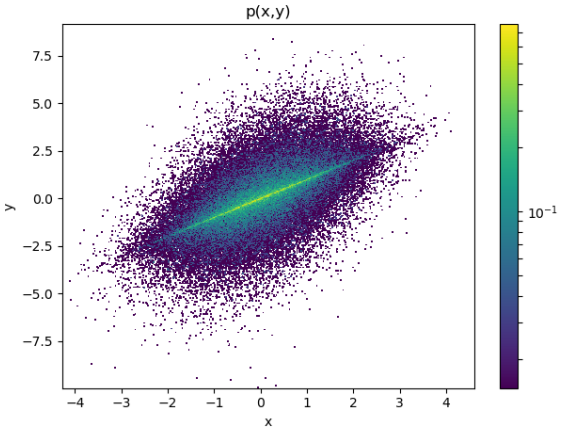


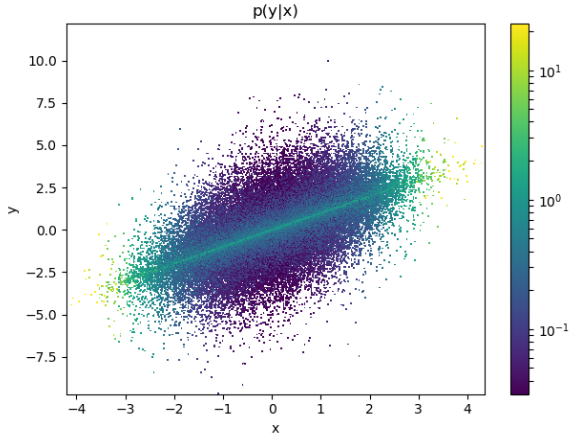


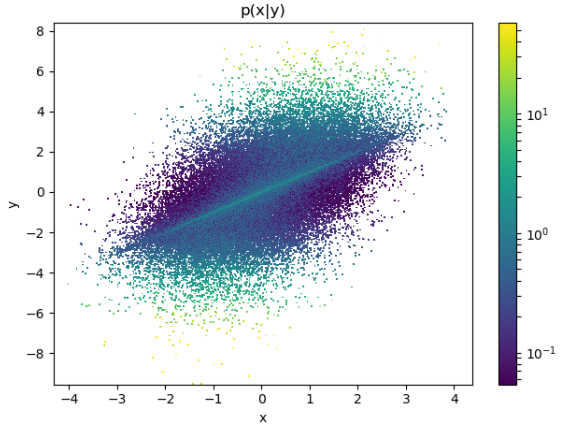


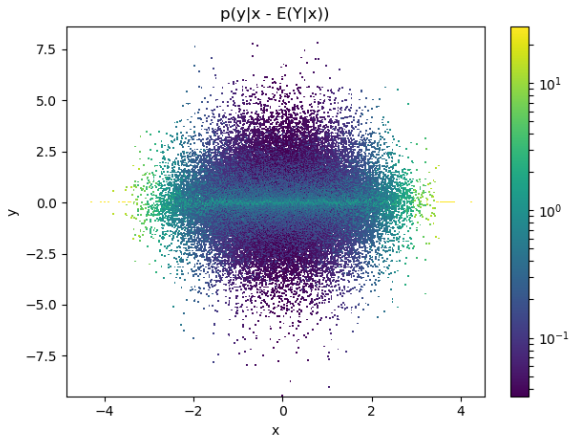


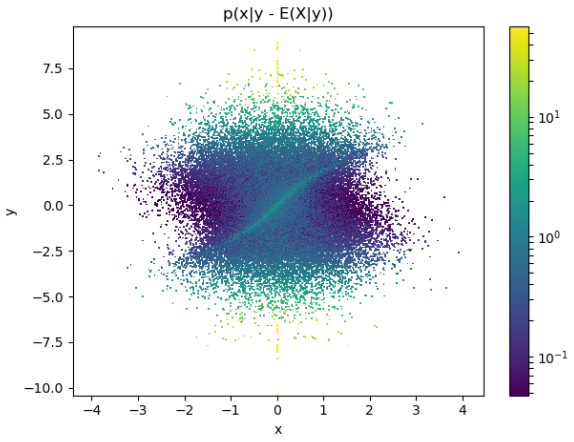
ج) داده‌های خطی با نویز غیرگاوسی، $b = 0$ و $q = 1.5$، $Y = X + sign(\mathcal{N}_y)|\mathcal{N}_y|^q$











سوال اول، بخش دوم:

(ب) در این بخش می‌خواهیم تاثیر غیرخطی بودن و غیرگاوسی بودن در تشخیص جهت علّی را بررسی کنیم. با استفاده از مدل بالا ۳۰۰ نمونه تولید کنید. سپس با استفاده از یک رگرسیون غیر خطی مانند SVR مدل را برازش کنید. اگر تابع برازش شده را $\hat{f}$ بنامیم و

$$\hat{n} = y - \hat{f}(x)$$

آنگاه اگر باقی‌مانده $\hat{n}$ بدست آمده مستقل از x باشد می‌توان گفت جهت مدل درست تشخیص داده شده است. برای بررسی استقلال این نمونه‌ها از تست آماری HSIC استفاده کنید و با سطح اطمینان ۲٪ فرضیه صفر را رد کنید. این آزمایش را ۱۰۰ بار تکرار کنید و نمودار مربوط به درصد پذیرش مدل در دو جهت را رسم کنید (کد مربوط به تست HSIC در اختیار شما قرار گرفته است.)

توابع استفاده شده در این بخش کارکرد زیر را دارند:

- تابع pureDist(x,y) : اثر تابعی x از y را بوسیله برازش SVR کم کرده و قسمت احتمالاً مستقل y از x را خروجی می‌دهد.
- تابع isIndependent(x, y, cLevel) : به کمک تابع pureDist قسمت خالص y را جدا کرده و تست استقلال HSIC را با حد cLevel انجام می‌دهد و نتیجه را اعلام می‌کند.

تابع اصلی این قسمت هم به صورت زیر است:

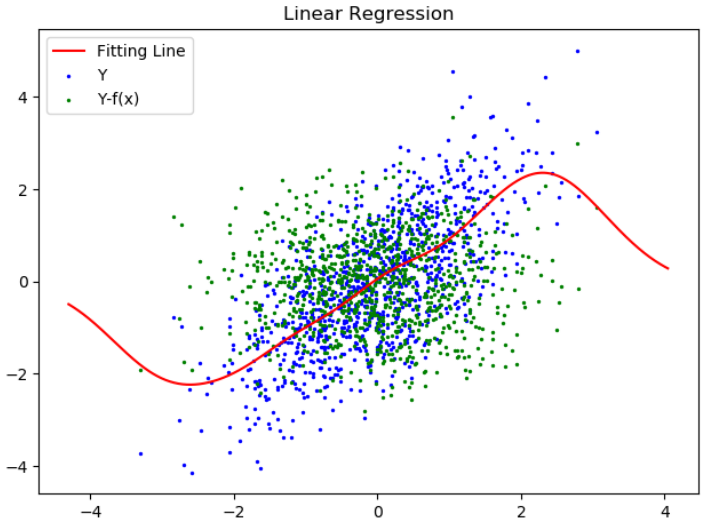
```
# Part 1 Session 1, X+X^3 Causal Test
def hsicTest():
    # Generating X Distribution: Nx = N(0, 1)
    meanNX = 0
    sigmaNX = 1
    yDependencyOfX = 0
    xDependencyOfY = 0
    for i in range(100):
        NX = np.random.normal(meanNX, sigmaNX, 300)
        # Calculating Y = X + bX^3 + sign(X)*|X|^q
        X = NX
        Y = yGen(X, b=0, q=2)
        if isIndependent(X, Y, cLevel=0.02):
            yDependencyOfX = yDependencyOfX + 1
        if isIndependent(Y, X, cLevel=0.02):
            xDependencyOfY = xDependencyOfY + 1
    print "Direction 1 Dependency = ", yDependencyOfX, \
        " ones, And Direction 2 Dependency = ", xDependencyOfY
```

که ۱۰۰ بار تست استقلال را در هر دو جهت انجام داده و نتیجه را اعلام می‌کند.

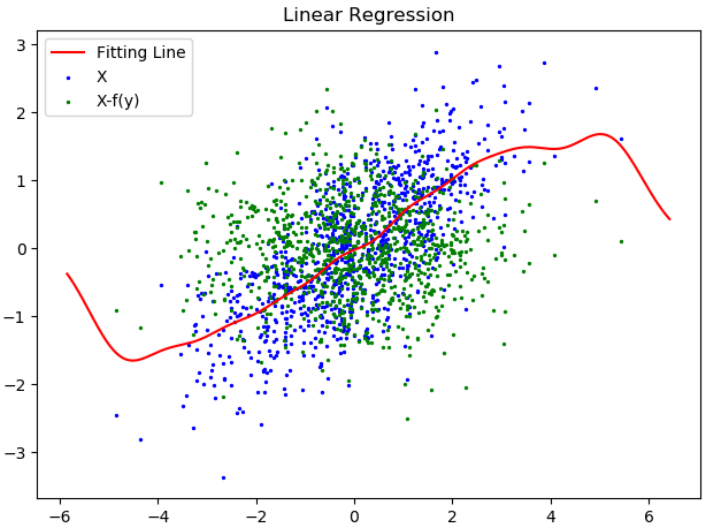
نکته: کرنل استفاده شده برای SVR ، کرنل RBF با C=1 است.

نتایج بدین صورت اند (نمودارها به جای ۳۰۰ داده به ۱۰۰۰ داده رسم شده اند):

الف) داده‌های خطی با نویز گاوسی، $b = 0$ و $q = 1$، $Y = X + \mathcal{N}_y$



برازش y روی x



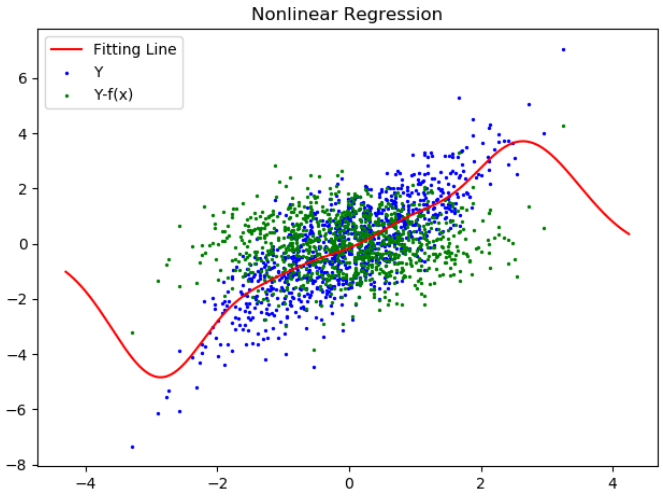
برازش x روی y

نتیجه روی ۱۰۰ آزمایش:

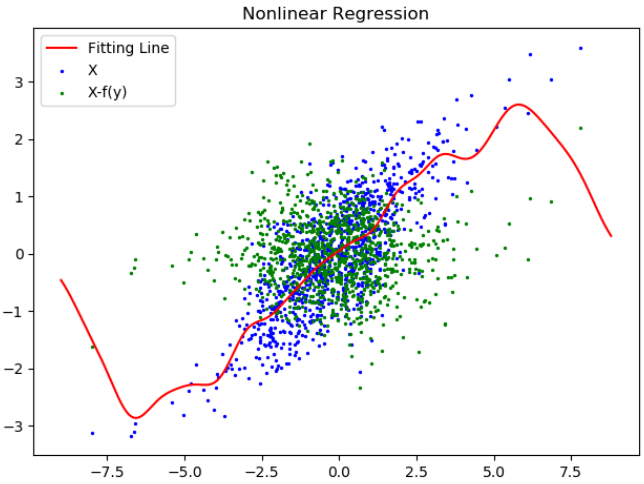
```
Y on X Direction Dependencies =  100 ones, And X on Y Direction Dependencies =  100 ones

Process finished with exit code 0
```

که نشان می‌دهد در حالت خطی، هر دو جهت از هم مستقل اند (منظور از هر عدد، تعداد **مستقل تشخیص داده شده ها** است).

ب) داده‌های غیرخطی با نویز گاوسی، $b = 0.5$ و $q = 1$، $Y = X + bX^3 + \mathcal{N}_y$



برازش y روی x



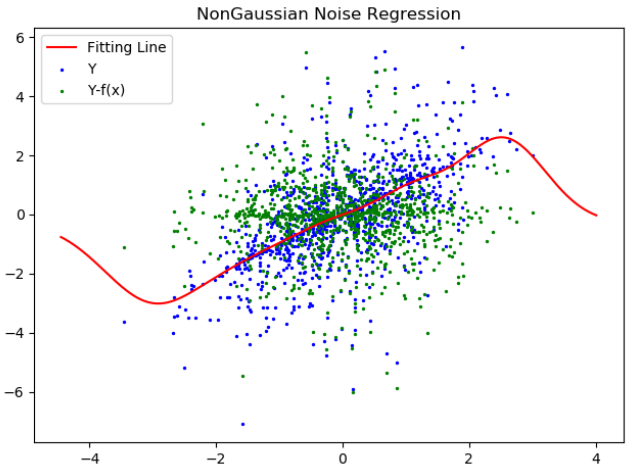
برازش x روی y

نتیجه روی ۱۰۰ آزمایش:

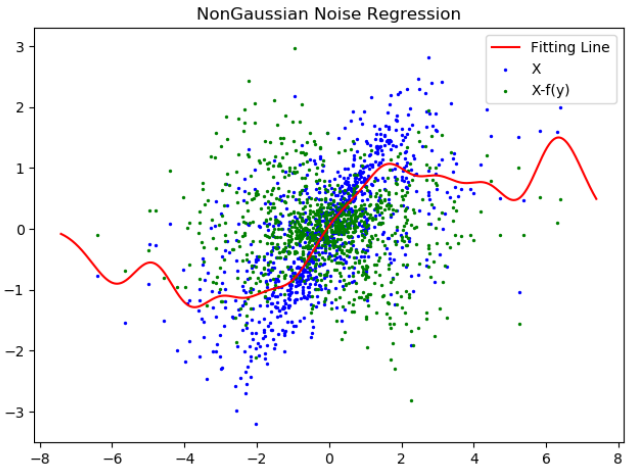
```
Y on X Direction Dependencies =  100 ones, And X on Y Direction Dependencies =  5 ones

Process finished with exit code 0
```

که نشان می‌دهد در حالت غیرخطی، همیشه Y از X مستقل است و در تعداد محدودی حالات (در اینجا ۵ درصد برای b=0.5) X نیز از Y مستقل تشخیص داده می‌شود.

ج) داده‌های خطی با نویز غیرگاوسی، $b = 0$ و $q = 1.5$، $Y = X + sign(\mathcal{N}_y)|\mathcal{N}_y|^q$



برازش y روی x



برازش x روی y

نتیجه روی ۱۰۰ آزمایش:

```
Y on X Direction Dependencies =  100 ones, And X on Y Direction Dependencies =  32 ones

Process finished with exit code 0
```

و برای q=2 :

```
Y on X Direction Dependencies =  100 ones, And X on Y Direction Dependencies =  2 ones

Process finished with exit code 0
```

که نشان می‌دهد در حالت نویز گاوسی، همیشه Y از X مستقل است و درحالت برعکس نیز، هرچه q بزرگتر باشد، X از Y نامستقل‌تر تشخیص داده می‌شود.

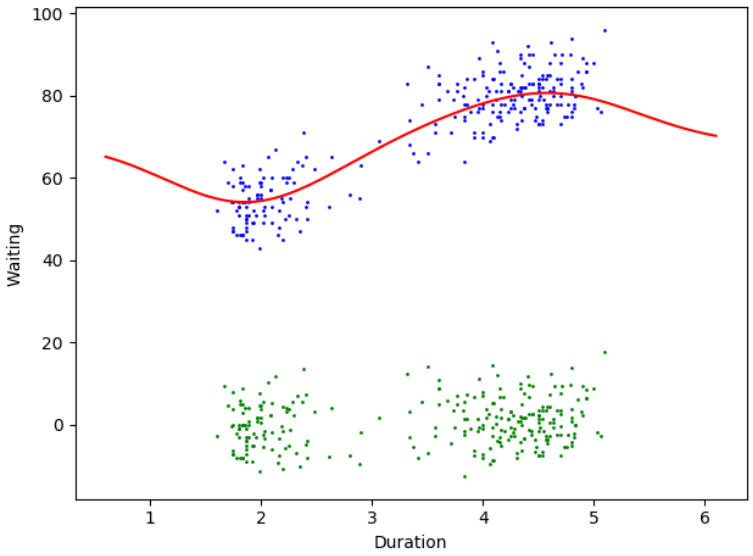
سوال اول، بخش سوم:

(ج) در این بخش قصد این را داریم که آزمون بالا را روی دو دسته داده دنیای واقعی بررسی کنیم:

- مجموعه اول[1] حاوی تعدادی نمونه از مدت زمان فوران و فاصله از فوران قبلی آبفشان است.
- مجموعه دوم[2] حاوی تعدادی نمونه از ویژگی‌های فیزیکی یک نوع صدف است. اطلاعات مربوط به این داده در این لینک[3] قرار دارد.

دادهها را از لینکهای داده شده دانلود کنید. سپس در مجموعه داده اول سعی کنید جهت صحیح رابطه‌ی علّی را میان مدت زمان فوران و فاصله فوران پیدا کنید. در مجموعه داده دوم سعی کنید جهت صحیح رابطه‌ی علّی میان تعداد حلقه‌ها و طول صدف را بدست آورید.

قسمت اول: رابطه زمان فوران و فاصله از فوران قبلی



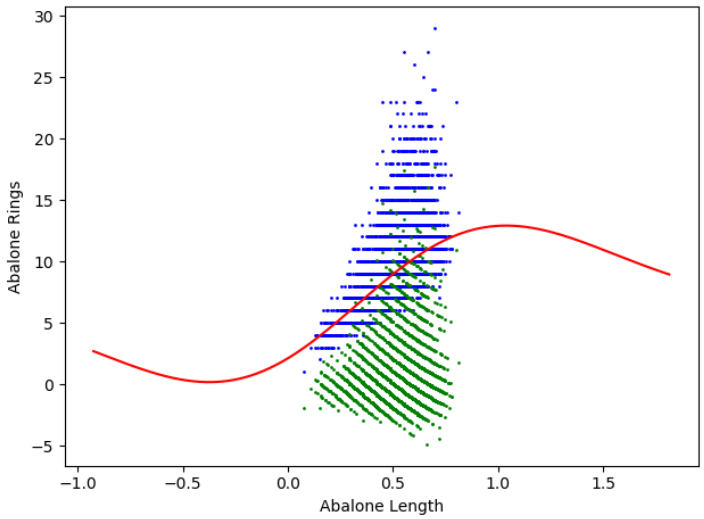
تابع استفاده شده در این بخش اینگونه است:

```
# Part 1 Session 2, Eruption Causal Test
def eruptionTest():
    fileName = "eruptions.csv"
    eruptionsData = pd.read_csv(fileName)
    eruptionDuration = np.array(eruptionsData['eruptions'])
    eruptionWaiting = np.array(eruptionsData['waiting'])
    dir1 = isIndependent(eruptionDuration, eruptionWaiting, 0.02)
    dir2 = isIndependent(eruptionWaiting, eruptionDuration, 0.02)
    if dir1 & ~dir2:
        print "Waiting is Cause of Duration"
    elif ~dir1 & dir2:
        print "Duration is Cause of Waiting"
    else:
        print "Undefined Causality"
```

و نتیجه همیشه به صورت زیر است:

```
Waiting is Cause of Duration

Process finished with exit code 0
```

قسمت دوم: رابطه تعداد حلقه و طول صدف



تابع استفاده شده در این بخش اینگونه است:

```
# Part 1 Session 3, Abalone Causal Test
def abaloneTest():
    fileName = "abalone.csv"
    abaloneColumns = ['Sex', 'Length', 'Diameter', 'Height', 'Whole weight',
                      'Shucked weight', 'Viscera weight', 'Shell weight', 'Rings']
    abaloneData = pd.read_csv(fileName, names=abaloneColumns)
    abaloneLength = np.array(abaloneData['Length'])
    abaloneRings = np.array(abaloneData['Rings'])
    dir1 = isIndependent(abaloneLength, abaloneRings, 0.02)
    dir2 = isIndependent(abaloneRings, abaloneLength, 0.02)
    if dir1 & ~dir2:
        print "Length is Cause of Rings"
    elif ~dir1 & dir2:
        print "Rings is Cause of Length"
    else:
        print "Undefined Causality"
```

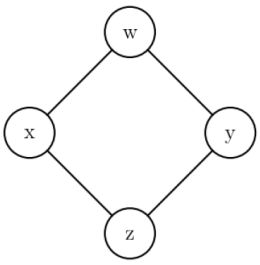
و نتیجه بدین صورت است، که چون هیچ جهتی حد آستانه را رد نمی‌کند اعداد را با جزییات بیشتری اعلام می‌کنیم:

```
TestStat= 3.7519056337833474 , And Threshold= 0.6332036484250122  So Variables are:  Dependent
TestStat= 4.803672439297263 , And Threshold= 0.6176079801661507  So Variables are:  Dependent
Undefined Causality

Process finished with exit code 0
```

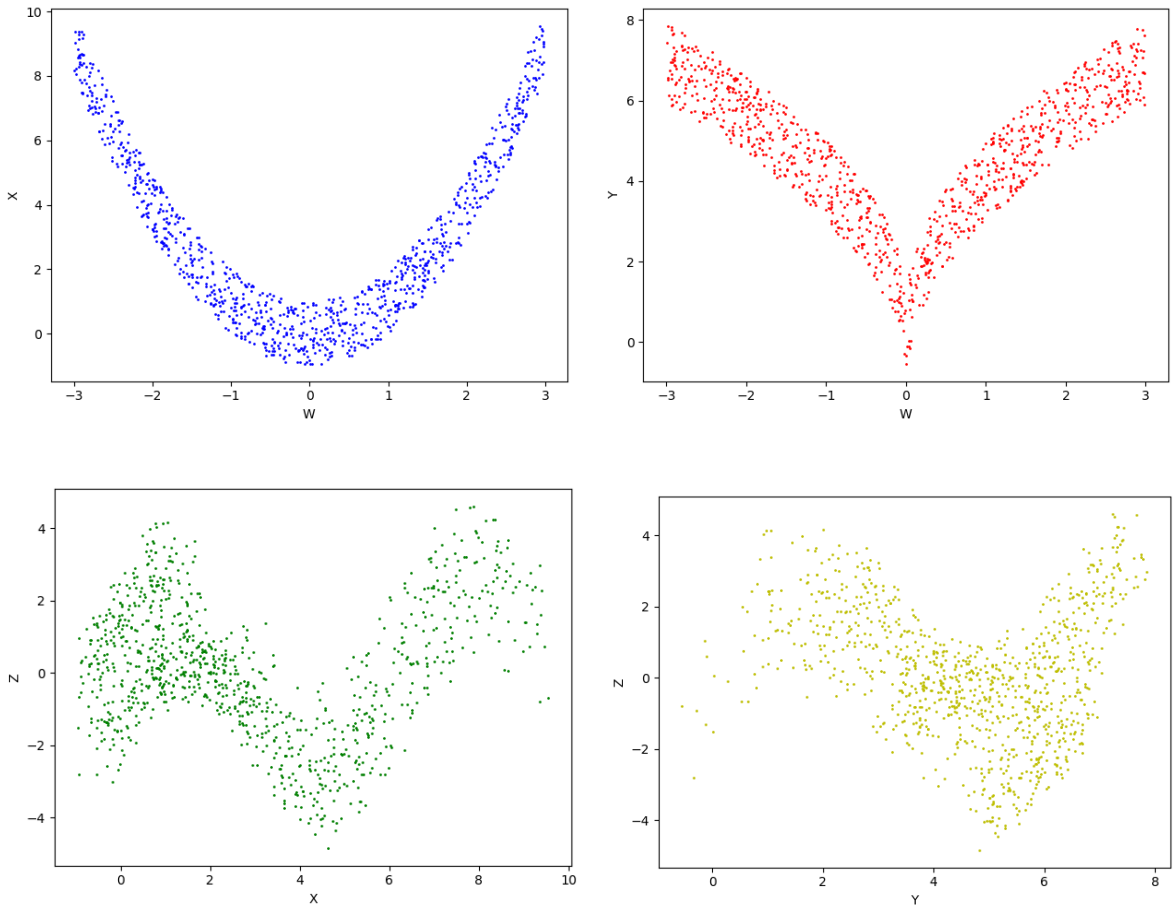
نتیجه می‌گیریم برای این دو متغیر، Structural Causal Model به شکل خوبی وجود ندارد. اما اگر مقید باشیم که حتما که جهت علت و معلولی تعیین کنیم، با توجه به آنکه حد آستانه هردو جهت حدود ۰.۶ است و معیار آماری اول مقدار کمتری دارد، احتمالا جهت درست‌تر این است که طول صدف تعیین کننده و علت تعداد حلقه‌های آن باشد.

سوال دوم:

۲. در این بخش قصد آن را داریم که روابط علّی را در یک گراف جهت‌دار بدون دور بررسی کنیم. مجموعه داده‌ای که در اختیارتان قرار گرفته توسط یک گراف با ساختار زیر تولید شده است. جهت درست روابط در این گراف جهت‌دار را حدس بزنید.



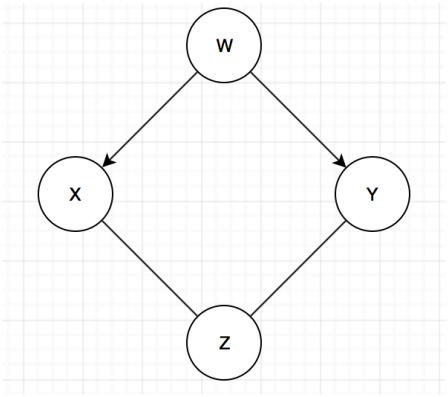
نمودار روابط یال‌ها به صورت زیر است:



و نتایج تست استقلال هیلبرت-اشمیت مطابق زیر است:

```
is W-->X ? : True
is X-->W ? : False
is W-->Y ? : True
is Y-->W ? : False
is X-->Z ? : False
is Z-->X ? : False
is Y-->Z ? : False
is Z-->Y ? : False

Process finished with exit code 0
```

پس نتیجه‌ای که تا اینجا می‌توانیم بگیریم به صورت زیر است:



حال، نتایج x-z و y-z را با دقت بیشتری بررسی می‌کنیم:

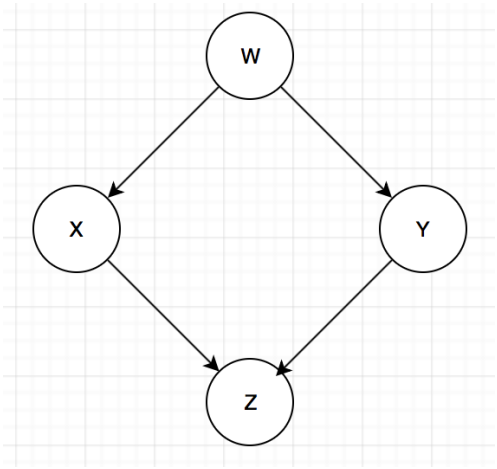
```
is X-->Z ? : TestStat= 2.3678198972238245 , And Threshold= 0.7191001126903502  So Variables are:  Dependent
False
is Z-->X ? : TestStat= 9.135229365543966 , And Threshold= 0.6854379085420673  So Variables are:  Dependent
False

Process finished with exit code 0
```

که تفاوت آشکار معیار آماری ۲ تا ۹، جهت x به z را به ما می‌دهد.

```
is Y-->Z ? : TestStat= 2.5425719002953544 , And Threshold= 0.7079095233614172  So Variables are:  Dependent
False
is Z-->Y ? : TestStat= 11.541812326070485 , And Threshold= 0.6917513112636148  So Variables are:  Dependent
False

Process finished with exit code 0
```

و نیز تفاوت زیاد معیار آماری ۲ تا ۱۱، جهت y به z را ارجحیت می‌دهد.

پس گراف جهت‌دار شده نهایی به صورت زیر می‌باشد:



که با شهود حاصل شده از نمودارها نیز برابری می‌کند.

..... پایان گزارش .....