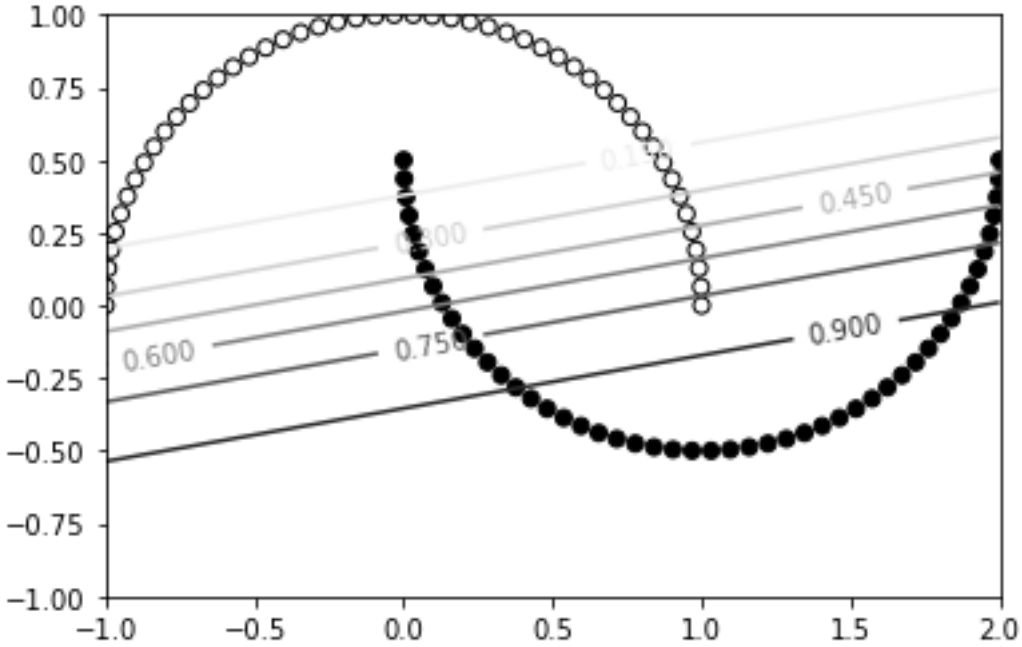
אתחול הפרמטרים

עד כה, כאשר הגדרנו מודל רשת נוירונים בעזרת PyTorch, לא הקדשנו מחשבה לערכי הפרמטרים הראשוניים: הם אותחלו באופן אקראי בעת הגדרת אובייקט הרשת, והיוו את נקודת ההתחלה עבור אלגוריתם האופטימיזציה הנבחר. עם זאת, כמו שכיוון התנועה וגודל הצעד של האלגוריתם חשובים, גם לנקודת ההתחלה השפעה מכרעת. בחירה שגויה של שיטת אתחול פרמטרים עלולה להוביל למספר בעיות.

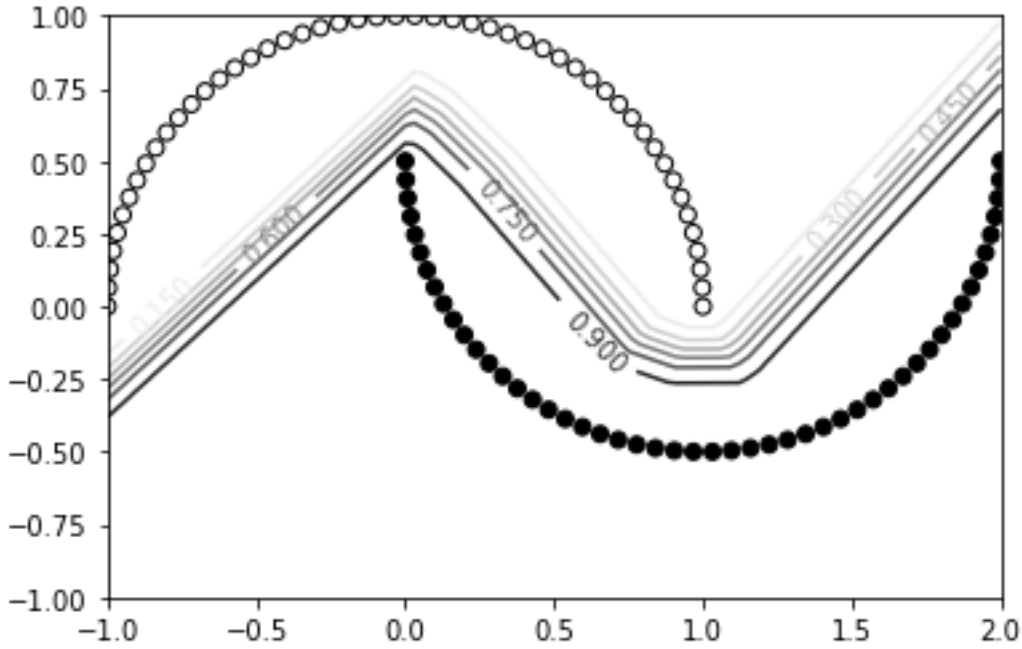
ראשית, נשים לב שאתחול הפרמטרים עבור שני נוירונים באותה שכבה באופן זהה יוביל לכך שהם יבצעו את אותו החישוב, ובפעפוע של השגיאה לאחור, הגרדיאנט אשר יחושב עבור פרמטרים אלו גם הוא יהיה זהה. תוצאת אתחול שכזה תהיה ששני הנוירונים יעודכנו בדיוק באותה צורה על ידי אלגוריתם האופטימיזציה, וגם באיטרציות הבאות ימשיכו לבצע את אותו החישוב. בכך הרשת תאבד מיכולת החישוב שלה. ראו בדוגמה הבאה, כיצד אנו מגדירים רשת עמוקה ומאתחלים אותה עם ערכי פרמטרים קבועים.

```
model=nn.Sequential(nn.Linear(2,10),
                    nn.ReLU(),
                    nn.Linear(10,10),
                    nn.ReLU(),
                    nn.Linear(10,1),
                    nn.Sigmoid())
for x in model.parameters():
  torch.nn.init.constant_(x,0.1)
```

אם נאמן רשת זו לסווג נקודות שחורות ולבנות (אשר יצרנו בעזרת הפונקציה `make_moons`), תתקבל התוצאה הבאה עבור הסתברויות הסיווג (נזכיר שההסתברות החזויה על ידי המודל לכך שנקודה נתונה היא שחורה כתובה על הקו העובר דרכה).



מאיור זה ניכר שבעוד שהרשת עמוקה, ובעלת מספר רב של נוירונים, אותה תוצאה היתה מתקבלת גם בעזרת מודל סיווג של נוירון יחיד. לשם השוואה, התוצאה המתקבלת מהאימון לאחר שימוש בפרמטרים המאותחלים כברירת מחדל בעת הגדרת הרשת מאוירת להלן, ומאיור זה ניכר שמורכבות הרשת באה לידי ביטוי בסיווג מוצלח יותר.



על כן, הדרישה הראשונה שלנו מאסטרטגיית האתחול תהיה "**שבירת הסימטריה**" (symmetry breaking) – נרצה שהפרמטרים יאותחלו כך שכל נוירון יבצע חישוב אחר ובהתאם יתעדכנו באופן שונה עם איטרציות אלגוריתם האופטימיזציה. בכדי לעמוד בדרישה זו, נאתחל את הפרמטרים באקראי.

אתחול אקראי אינו הפתרון לכל הבעיות, שכן פרמטרים קטנים מדי יובילו לערכי נגזרות קטנים מדי, ואלגוריתם האופטימיזציה לא יזוז מנקודת ההתחלה. תופעה זו נקראת "הגרדיאנט הנעלם" (vanishing gradient) וניתקל בה בהמשך, גם בהקשר של ארכיטקטורות רשת מסוימות. קיימת גם הבעיה ההפוכה: אתחול ערכי פרמטרים גדולים מדי (בערכם המוחלט) יוביל לערכי נגזרות גדולים מדי והאלגוריתם יתבדר, זו תופעת "הגרדיאנט המתפוצץ" (exploding gradient). לשתי בעיות אלו קיימים מספר פתרונות, ומקורן לאו דווקא רק מאתחול פרמטרים שגוי. עם זאת, על ידי בחירת סדר גודל נכון לפרמטרים ההתחלתיים ניתן לעתים להתחמק מהן. הכללים המקובלים בשימוש נפוץ קובעים את **השונות** של מחולל המספרים האקראיים עבור הפרמטרים של שכבה מסויימת כך שבו זמנית במעבר קדימה (בעת הזנת הנתונים ברשת) ובמעבר אחורה (בעת פעפוע הגרדיאנט), הערכים המחושבים ישמרו על אותו סדר גודל. הקו המנחה מאחורי שיטות אלו הוא שככל אשר לשכבה מסוימת יש יותר משתני קלט ופלט, כך יש לאתחל את הפרמטרים עם שונות קטנה יותר.

שיקולים אלו נלקחו בחשבון בעת תכנון הבנאים של המודולים המובנים ב-PyTorch: לכל מודול/ שכבה בעלת פרמטרים יש מתודה בשם `reset_parameters()` אשר מאתחלת את הפרמטרים לפי הכלל המתאים. למשל בשכבה ליניארית, `nn.Linear`, הפרמטרים יאותחלו באקראי לפי התפלגות אחידה בקטע $\left[-\frac{1}{\sqrt{K_{in}}},\frac{1}{\sqrt{K_{in}}}\right]$, כאשר K_{in} הוא מימד הקלט של השכבה.

שאלות לתרגול

1.

א. אתחלו את פרמטרי הרשת המוגדרת לעיל לפי שיטת Xavier : הפרמטרים יאותחלו באקראי לפי התפלגות אחידה בקטע $\left[-\frac{\sqrt{6}}{\sqrt{K_{in}+K_{out}}},\frac{\sqrt{6}}{\sqrt{K_{in}+K_{out}}}\right]$, כאשר K_{out}

הוא מימד פלט השכבה, ו- K_{in} הוא מימד הקלט. **רמז:** השתמשו בפונקציה `torch.nn.init.uniform_`.

ב. אמנו את הרשת לסיווג הנקודות השחורות והלבנות אשר נוצרו בעזרת הפונקציה `make_moons`.

ג. השוו את התוצאות לרשת המאותחלת לפי ברירת המחדל.