שיפור האלגוריתם

המטרה: לבנות אלגוריתם שיוכל ליצור פונקציית מגמה בכל חזקה שנרצה.

לשם כך, נגדיר את פונקציית המגמה כ**פולינום**:

כך ש היא סדרה של המקדמים. כך נוכל ליצור מערך מקדמים ולהשתמש בו במקום בערכי m ו b שלנו.

אז מתקיים:

וכמו כן,

כאשר n היא החזקה הגבוהה ביותר ו m הוא מספר הנתונים ב x.

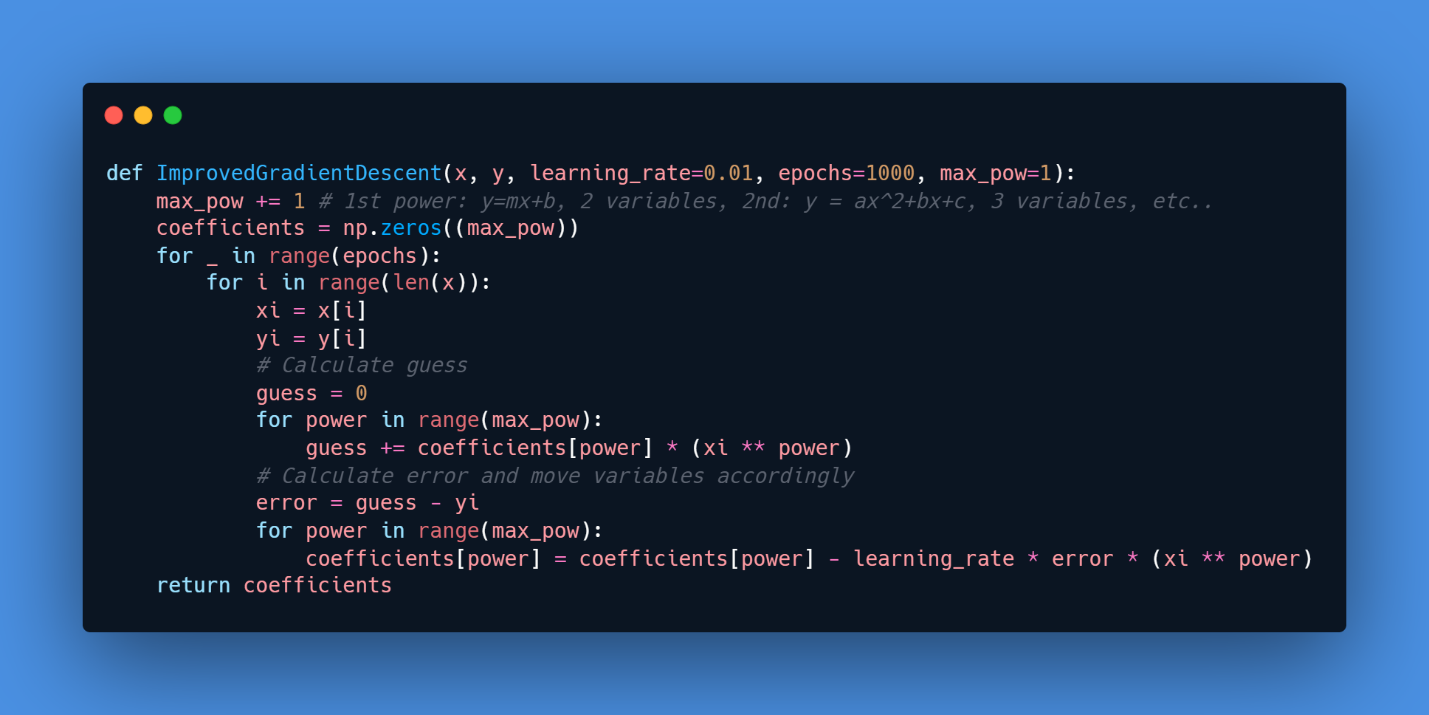
נגזור את הפונקציה עבור . אז כל מה שהוא לא – כל שאר איברי הסדרה a, וגם כל משתנה אחר, הוא קבוע.

בדומה לפתרון ה Gradient Descentהמקורי, נגזור כל איבר בסכום (החיצוני) בנפרד.

בדומה לפתרון המקורי, הכפל ב 2 זניח.

לכן עבור כל איבר בסדרה, יש לבצע:

מימוש האלגוריתם בפייתון



בדיקה והשוואה

בחרתי לרחם על המחשב שלי הפעם ולבחור מספר נתונים קטן, אז בחרתי את ההשוואה בין הוצאות השיווק של בנטון למכירות שלהם – עשרה נתונים בסך הכל.

לצורך בקרה, יצרתי גם קו מגמה בשיטה הישנה והשוויתי את ערכי ה Cost שלו.

הפתרון המלא: <https://github.com/itays123/DataScience/blob/main/Z1%20Grad-Descent-Math/improvement.py>

לאחר ששיחקתי קצת עם ערכי learning\_rate ו epochs, כאשר epochs תלוי בחזקה, הגעתי לתוצאות מרשימות:

|  |  |
| --- | --- |
| החזקה | Cost |
| בקרה | 188.03929200803836 |
| 1 | 189.76126351922682 |
| 2 | 116.19233143009637 |
| 3 | 98.23432062528778 |
| 4 | 94.57164467036893 |
| 5 | 86.0313261409806 |

כבר בחזקה השנייה רואים שיפור משמעותי לעומת החזקה הראשונה, אבל בחזקות גבוהות יותר הצלחתי לצמצם ביותר מחצי את ערך ה Cost!