Seminarski

Simbolicka regresija

autori: mi19108, mi19255

Uvod

Simbolicka regresija (SR) je tip regresione analize koja trazi matematičke izraze kako bi našla model koji se najbolje uklapa u dati set podataka.

Naš pristup se bazira na konceptu Expression Tree-ja,
kod kojeg se izgradi stablo sačinjeno od objekata tipa Node, čija se definicija može naći ispod.

```
class Node:
    def __init__(self, operation=None, left=None, right=None, value=None):
        self.operation = operation
        self.left = left
        self.right = right
        self.value = value
```

Ovde možemo uočiti da klasa Node sadrži četiri vrednosti, tako da može biti ili operacija gde se vrednosti čuvaju u levom i desnom podstablu (tip Node), ili može imati vrednost na koju se odgovara nadoperacijom.

Pristup

Naš izabrani pristup se svodi na genetsko programiranje za nalaženje najbolje populacije kao sam odgovor na pitanje, i biranje najbolje jedinke iz te populacije kao funkciju kojom će tačke na grafiku biti opisane.

Biranje početnih podataka

```
X = np.random.rand(100, 1)
y = X + np.random.randn(100, 1)
```

Početni podaci se nalaze u vidu tačaka izabranih pseudo nasumično, gde su tačke na x osi ključevi, a na y osi vrednosti na date ključeve (parametre).

Algoritam

Opredelili smo se za genetski algoritam koji ćemo ručno implementirati. Jedine dodatne biblioteke koje su u projekat ubačene su:

1. numpy
2. random
3. math
4. matplotlib

Početak

Na početku se generišu izrazi nasumično:

```
def generate_expression(depth):
    if depth == 0:
        return Node(value=random.uniform(-100, 100))
    else:
        op = random.choice(["+", "-", "*", "/", "cos+",
                            "cos-", "cos*", "cos/", "sin+",
                            "sin-", "sin*", "sin/"])
        left = generate_expression(depth - 1)
        right = generate_expression(depth - 1)
        return Node(operation=op, left=left, right=right)
```

Merenje greške

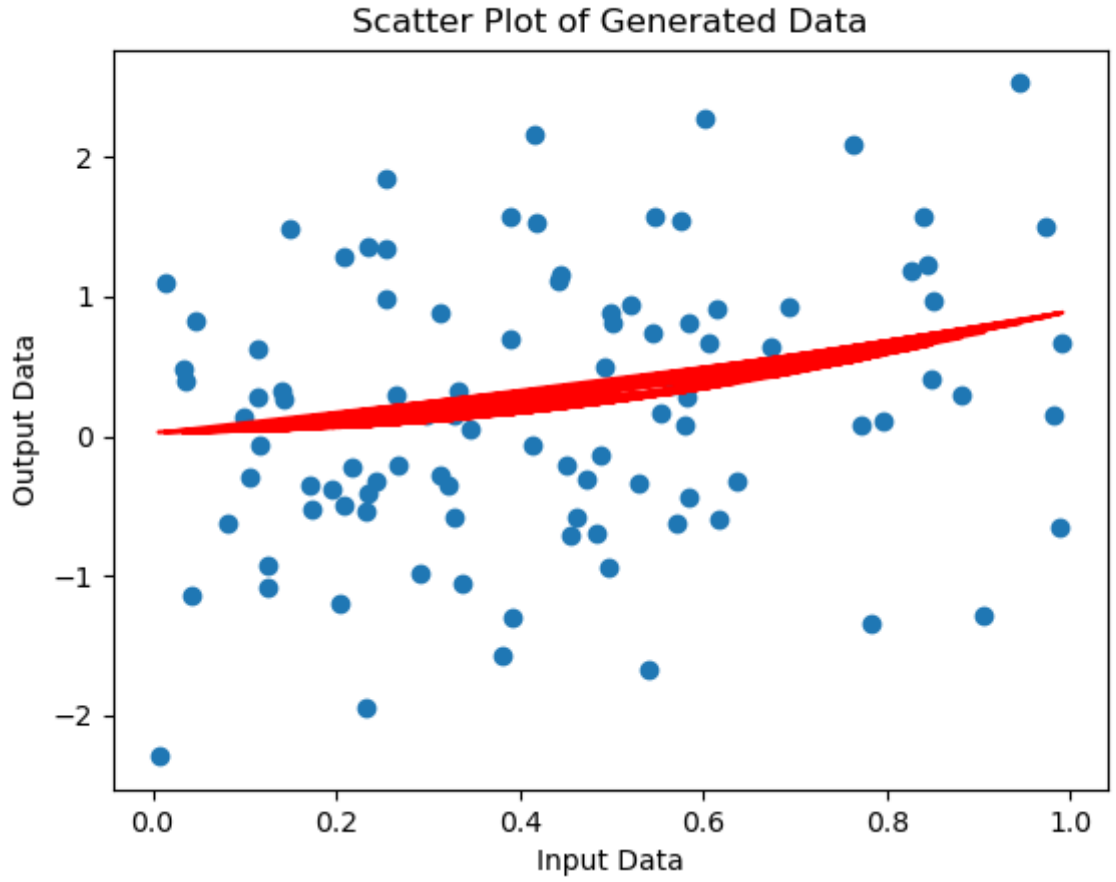
Za projekat je korišćena MSE metrika za fitness funkciju:

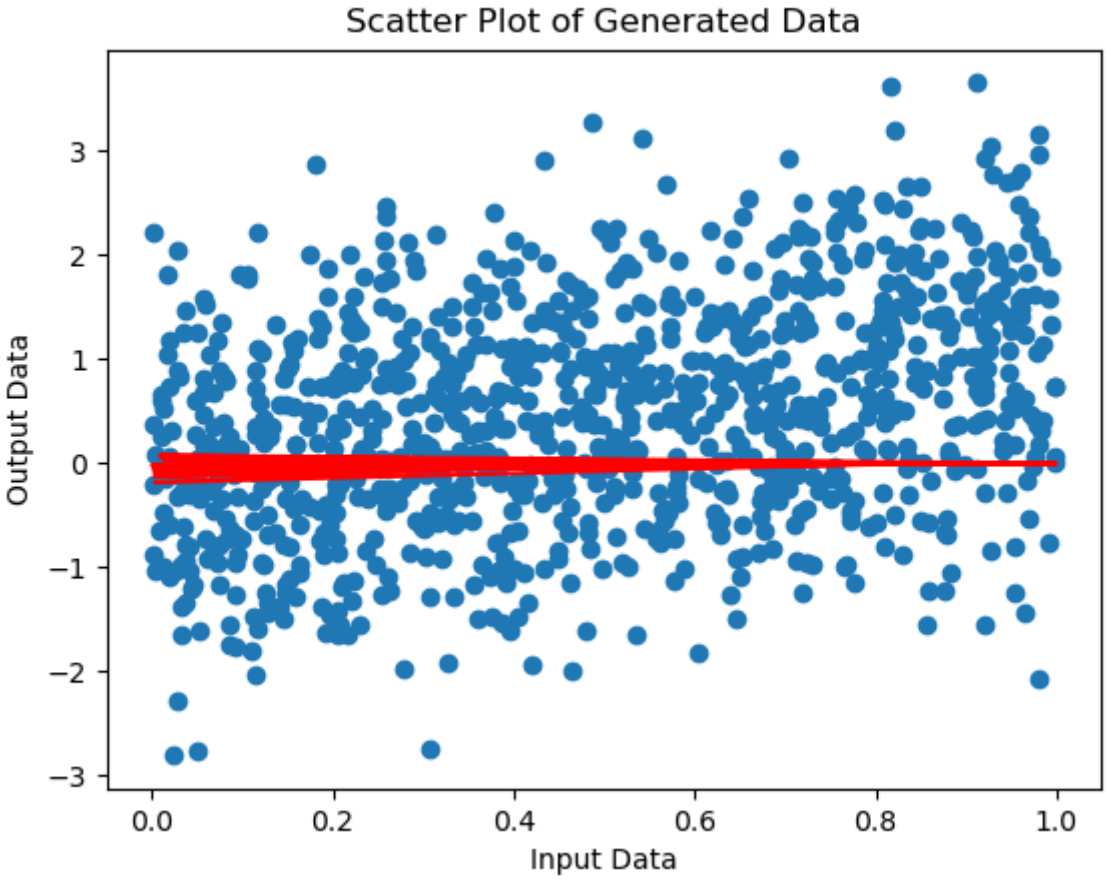
```
def fitness(individual, X, y):
    predictions = [individual.evaluate(x) for x in X]
    for i in range(len(predictions)):
        if predictions[i] == None:
            predictions[i] = 0.0

    mse = ((predictions - y) ** 2).mean()
    return 1 / (mse + 1e-9)  # Dodajemo ovu malu konstantu
```

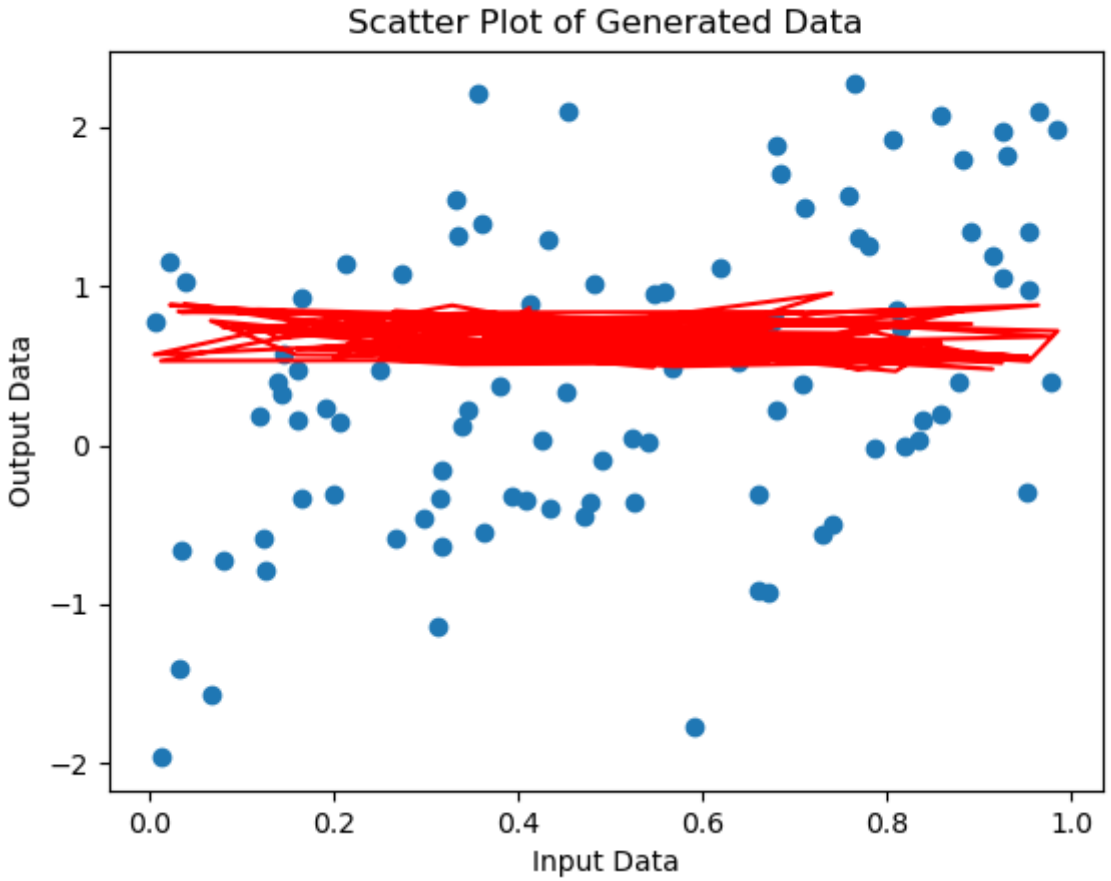
Inicijalni grafici

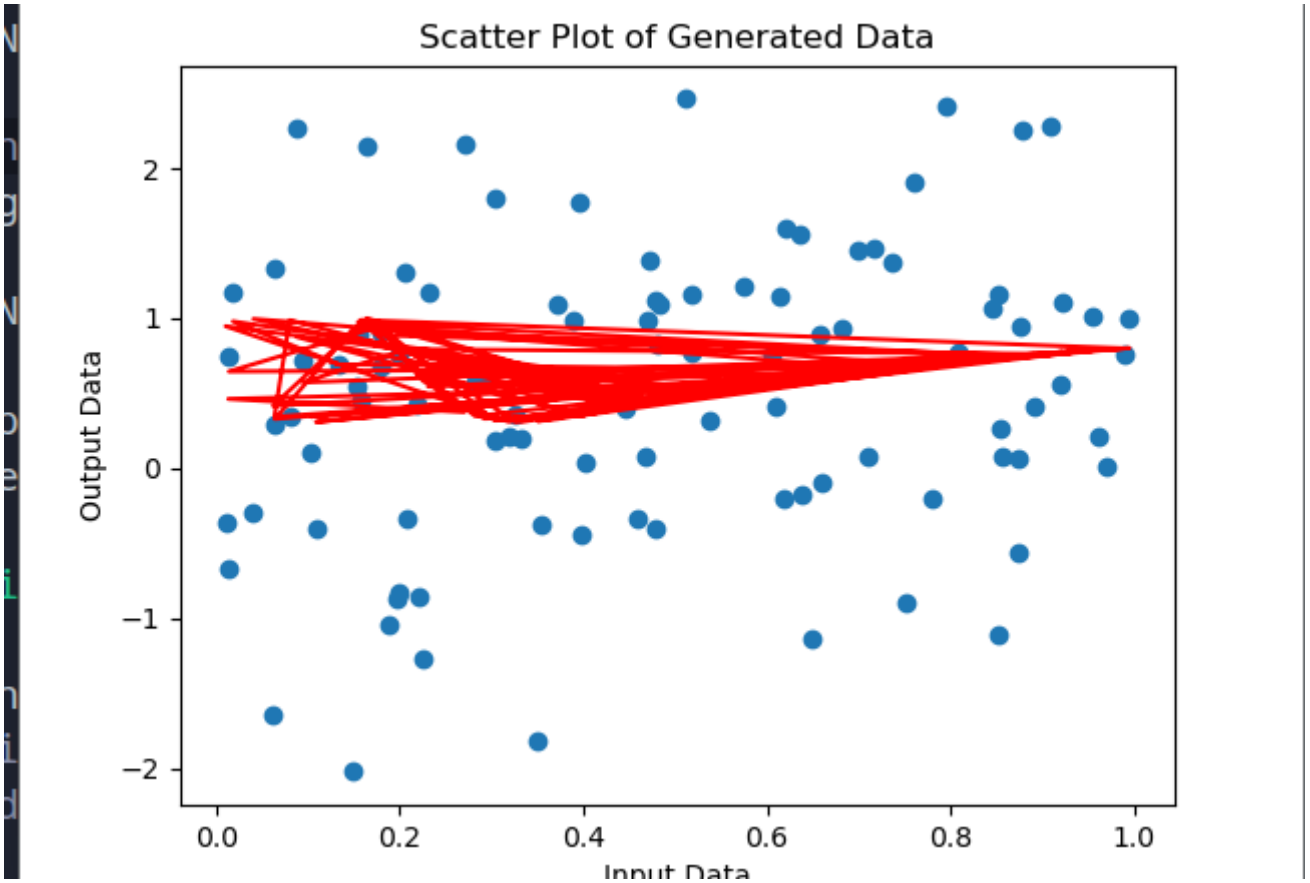
Na inicijalnim graficima je prikazano ponašanje sa manjim brojem funkcija, i nasumično odabranim vrednostima za broj izraza, veličinu izraza, dubinu izraza, veličinu jedne generacije, šansu za mutaciju:

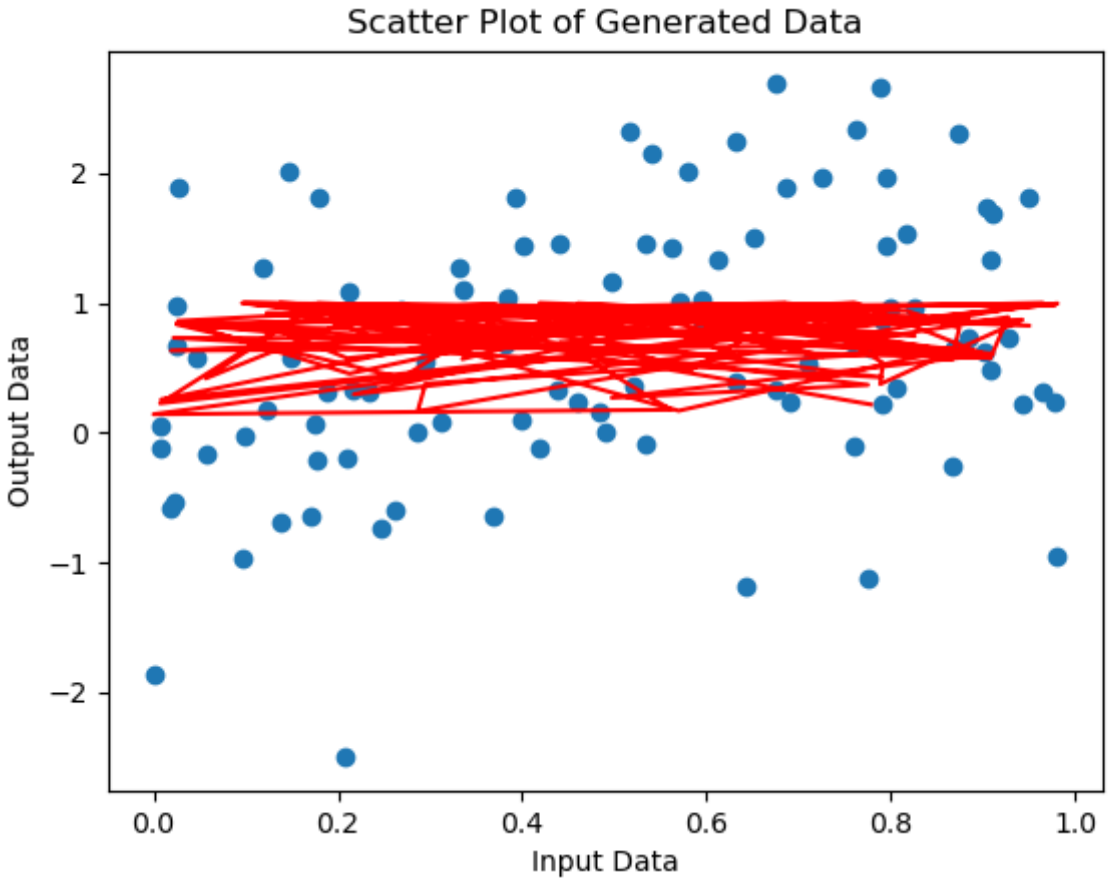




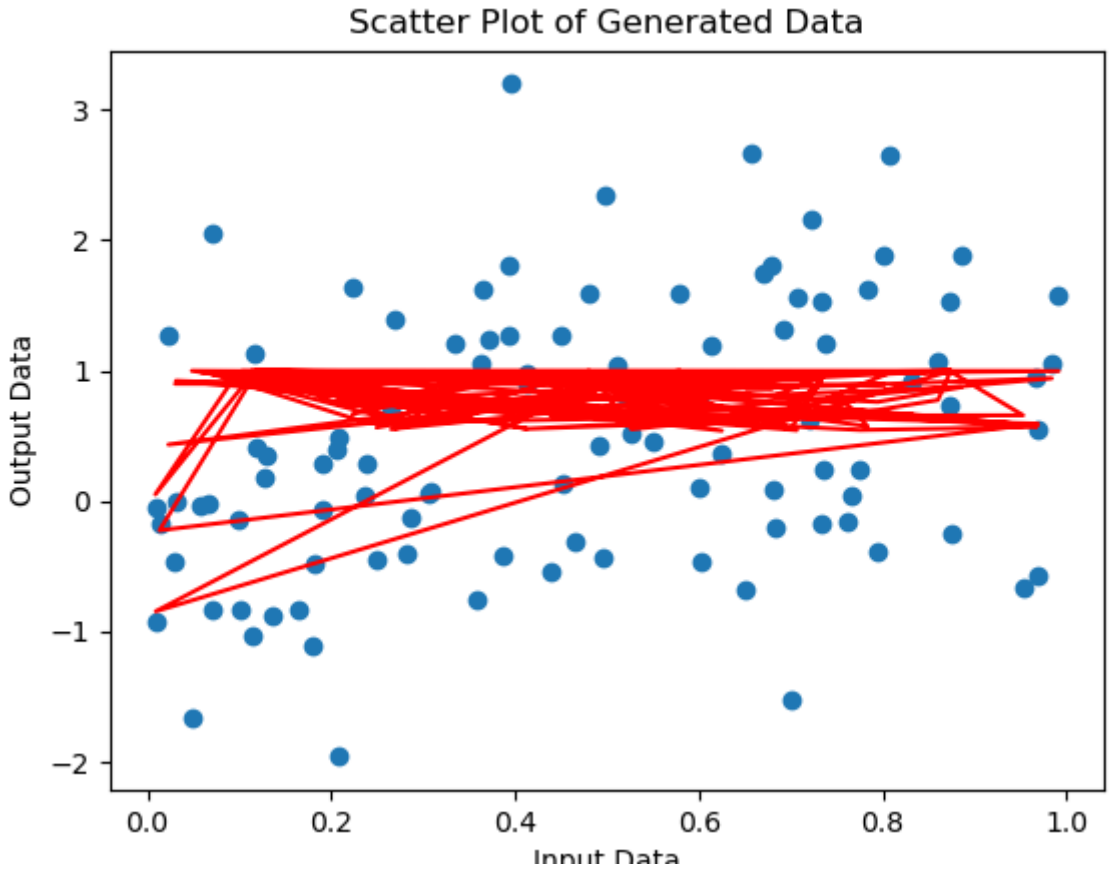
Daljim menjanjem ovih vrednosti, kao i dodavanjem trigonometrijskih funkcija, dobijeni su bolji rezultati:


Scatter Plot of Generated Data
Output Data
Input Data
2
1
0
−1
−2
0.0
0.2
0.4
0.6
0.8
1.0


Scatter Plot of Generated Data
Output Data
Input Data
2
1
0
−1
−2
0.0
0.2
0.4
0.6
0.8
1.0



Posle dodatnog menjanja funkcija, rezulat je počeo da bolje prijanja uz same vrednosti.



Zakljucak

Zbog velike količine rekurzije, kao i zbog toga što se programske instrukcije smenjuju na procesoru bez podrške grafičke kartice, količina simulacija je ograničena sistemom na kome se pokreće. Dalje metode za unapređenje bi se svodile na uvođenje procesuiranja aritmetičkih operacija na grafičkom procesoru.

Na eksperimentalan način su nađeni brojevi poput veličine populacije, dubine izraza i sl.

Dalje optimizacije u vidu pretprocesiranja bi mogle dovesti do bržeg završetka izvršavanja programa, ali ne u dovoljnoj količini.

Sve simulacije su obavljene na Intel Pentiumu iz 2009. godine.

Reference

1. https://proceedings.mlr.press/v139/biggio21a.html
2. Mark J. Willis; Hugo G. Hiden; Ben McKay; Gary A. Montague; Peter Marenbach (1997). "Genetic programming: An introduction and survey of applications"
3. Askhat Diveev, Elizaveta Shmalko Machine Learning Control by Symbolic Regression [1 ed.]