Dokumentacja Projektu

Optymalizacja Hiperparametrów XGBoost przy użyciu Algorytmu Ewolucyjnego

Mikołaj Rożek, Maksymilian Bilski

19 listopada 2025

1 Wprowadzenie

Celem projektu jest znalezienie optymalnego zestawu hiperparametrów dla modelu klasyfikacyjnego XGBoost (Extreme Gradient Boosting) w celu maksymalizacji jego skuteczności. Jako problem testowy wykorzystany zostanie zbiór danych Titanic, dostępny na platformie Kaggle: Titanic Dataset.

Do rozwiązania problemu optymalizacji zostanie wykorzystany algorytm ewolucyjny, który będzie przeszukiwał przestrzeń możliwych hiperparametrów w sposób inspirowany ewolucją biologiczną.

2 Proponowana Metodologia

Poniżej opisane zostaną kluczowe komponenty implementowanego algorytmu ewolucyjnego.

2.1 Osobnik populacji

Pojedynczy osobnik w populacji będzie reprezentował kompletny zestaw hiperparametrów dla modelu XGBoost. Każda cecha osobnika będzie odpowiadała jednemu hiperparametrowi. Optymalizowane parametry to:

- **eta (learning rate):** Szybkość uczenia modelu.
- **max_depth:** Maksymalna głębokość drzewa decyzyjnego.
- **subsample:** Frakcja próbek uczących używana do trenowania każdego drzewa.
- **colsample_bytree:** Frakcja cech (kolumn) używana przy budowie każdego drzewa.
- **min_child_weight:** Minimalna suma wag instancji potrzebna w węźle liścia.
- **gamma:** Minimalna redukcja funkcji straty wymagana do wykonania podziału.
- **n_estimators:** Liczba drzew w modelu.

Każdy z tych parametrów będzie miał zdefiniowany zakres wartości (dyskretnych lub ciągłych), w obrębie którego będzie mógł być modyfikowany.

2.2 Populacja Początkowa

Algorytm rozpocznie działanie od stworzenia populacji początkowej, składającej się z ustalonej liczby losowych osobników. Wartości poszczególnych hiperparametrów dla każdego osobnika zostaną wylosowane niezależnie z ich predefiniowanych zakresów.

2.3 Funkcja Oceny (Fitness)

Jakość każdego osobnika w populacji będzie mierzona za pomocą funkcji oceny, nazywanej również funkcją fitness. W tym projekcie proces oceny będzie przebiegał następująco:

1. Zestaw danych zostanie podzielony na zbiór treningowy i testowy.
2. Model XGBoost będzie trenowany na zbiorze treningowym przy użyciu hiperparametrów z ocenianego osobnika.
3. Jakość wytrenowanego modelu zostanie ewaluowana na zbiorze testowym za pomocą metryki **F1-score**.

Wartość F1-score, będąca średnią harmoniczną precyzji i czułości, będzie stanowić wartość funkcji fitness danego osobnika. Wyższa wartość będzie oznaczać lepsze przystosowanie.

2.4 Selekcja

W celu wyłonienia osobników do reprodukcji, będzie tworzona **pula rodzicielska** (ang. mating pool) za pomocą **selekcji turniejowej (Tournament Selection)**. Proces ten będzie przebiegał następująco: aby wybrać pojedynczego rodzica do puli, z aktualnej populacji zostaną wylosowani 'k' osobników (gdzie 'k' to rozmiar turnieju). Ten z nich, który ma najwyższą wartość funkcji fitness, wygra turniej i zostanie dodany do puli rodzicielskiej. Operacja ta będzie powtarzana, aż pula rodzicielska osiągnie rozmiar równy rozmiarowi populacji. Taki mechanizm sprawi, że silniejsze osobniki będą miały statystycznie większą szansę na wielokrotne skopiowanie do puli rodzicielskiej.

2.5 Operatory Ewolucyjne

Nowe pokolenia osobników będą tworzone przy użyciu operatorów krzyżowania i mutacji.

2.5.1 Krzyżowanie (Crossover)

Po stworzeniu i potasowaniu puli rodzicielskiej, osobniki będą dobierane w pary. Dla każdej pary rodziców będzie podejmowana decyzja o krzyżowaniu na podstawie hiperparametru `crossover_prob`.

- Jeśli krzyżowanie będzie zachodzić (z prawdopodobieństwem `crossover_prob`), para rodziców stworzy **dwóch nowych osobników potomnych**, którzy trafią do populacji następnej generacji. Potomkowie odziedziczą cechy po rodzicach, np. poprzez losowy wybór cechy od jednego z rodziców dla każdego hiperparametru (krzyżowanie jednorodne).
- Jeśli krzyżowanie nie będzie zachodzić, obaj rodzice zostaną **bezpośrednio skopiowani** do populacji następnej generacji.

Taki mechanizm zagwarantuje, że rozmiar populacji pozostanie stały w każdym pokoleniu.

2.5.2 Mutacja (Mutation)

Każda cecha (hiperparametr) w każdym osobniku nowej populacji będzie miała szansę na mutację, określoną przez parametr `mutation_prob`. Jeśli dana cecha zostanie wylosowana do mutacji, sposób modyfikacji jej wartości będzie zależał od jej typu (ciągły lub dyskretny):

- **Dla parametrów ciągłych** (np. `eta`, `gamma`): Wartość cechy zostanie zmodyfikowana o niewielką, losową wartość z rozkładu normalnego. Nowa wartość zostanie następnie sprawdzona i ewentualnie przycięta, aby mieściła się w predefiniowanym, dopuszczalnym zakresie.

- **Dla parametrów dyskretnych** (np. `max_depth`): Zastosowana zostanie mutacja pełzająca (creep mutation). Do obecnej wartości cechy zostanie dodana losowa wartość z rozkładu normalnego, a wynik zostanie **zaokrąglony do najbliższej liczby całkowitej**. Podobnie jak w przypadku parametrów ciągłych, finalna wartość zostanie sprawdzona, aby nie wykraczała poza swój dopuszczalny zakres.

Takie zróżnicowane podejście gwarantuje, że zmutowane wartości hiperparametrów będą zawsze poprawne i dopasowane do ich natury, a jednocześnie wprowadzi nową informację do populacji, co pozwoli na eksplorację nowych obszarów przestrzeni rozwiązań.

2.6 Sukcesja

Ostatnim krokiem w każdej generacji będzie stworzenie nowej populacji na podstawie populacji rodzicielskiej i potomnej. W projekcie zostanie zastosowana strategia **sukcesji elitarnej**, która zagwarantuje, że najlepsze znalezione dotąd rozwiązania nie zostaną utracone. Proces ten, kontrolowany przez parametr `elite_size`, będzie przebiegał następująco:

1. `elite_size` najlepszych osobników z populacji rodzicielskiej zostanie bezpośrednio skopiowanych do nowej populacji.
2. Pozostałe `population_size - elite_size` miejsc w nowej populacji zostanie zapełnionych przez najlepszych osobników z nowo utworzonej populacji potomnej.

Dzięki temu ogólna wartość przystosowania populacji nie będzie mogła się pogorszyć z pokolenia na pokolenie pod względem najlepszego osobnika.

2.7 Parametry Algorytmu Ewolucyjnego

Działanie samego algorytmu ewolucyjnego będzie kontrolowane przez zestaw hiperparametrów, które będą ustalane przez użytkownika przed uruchomieniem procesu optymalizacji. Kluczowe parametry w tym projekcie to:

- **population_size:** Liczba osobników (zestawów hiperparametrów XGBoost) w każdej populacji.
- **number_of_generations:** Liczba pokoleń, przez które będzie ewoluowała populacja. Definiuje to, jak długo będzie działał algorytm.
- **crossover_prob:** Prawdopodobieństwo (wartość od 0.0 do 1.0), z jakim para rodziców zostanie poddana operacji krzyżowania w celu stworzenia potomstwa.
- **mutation_prob:** Prawdopodobieństwo (wartość od 0.0 do 1.0), z jakim każda pojedyncza cecha (hiperparametr) w osobniku potomnym ulegnie mutacji.
- **tournament_size:** Liczba osobników biorących udział w pojedynczym turnieju podczas selekcji turniejowej.
- **elite_size:** Liczba najlepszych osobników z populacji rodzicielskiej, która zostanie bezpośrednio skopiowana do następnego pokolenia.

3 Proces Ewolucji i Ewaluacja

Algorytm będzie przebiegał iteracyjnie przez zadaną liczbę pokoleń. W każdej iteracji (pokoleniu) będą wykonywane następujące kroki:

1. Ocena wartości fitness każdego osobnika w bieżącej populacji.

2. Stworzenie populacji potomnej poprzez selekcję, krzyżowanie i mutację.

3. Zastosowanie sukcesji elitarnej w celu stworzenia następnego pokolenia z populacji rodzicielskiej i potomnej.

W trakcie działania algorytmu będzie śledzona najwyższa wartość fitness w każdym pokoleniu. Po zakończeniu wszystkich iteracji, najlepszy znaleziony w całym procesie ewolucji osobnik (zestaw hiperparametrów) zostanie uznany za rozwiązanie problemu optymalizacji.

Ostateczna skuteczność znalezionych parametrów zostanie zweryfikowana poprzez ponowne wytrenowanie modelu XGBoost na zbiorze treningowym i ocenę jego metryk (F1-score oraz Accuracy) na zbiorze testowym, który nie będzie używany w procesie ewolucji.