Raport z projektu

06-DUMALI0 Semestr zimowy 2024/2025

# Cel projektu

Celem projektu było stworzenie modelu, który przewiduje ceny domów na podstawie danych dotyczących cech nieruchomości i jej otoczenia. Projekt oparto na danych z konkursu **House Prices - Advanced Regression Techniques** na platformie Kaggle. Modele zostały ocenione pod kątem zdolności do przewidywania zmiennej docelowej SalePrice.

# Dane

Dane pochodzą z konkursu **House Prices - Advanced Regression Techniques** ([link do konkursu](https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/overview)).

* **Zbiór uczący** zawierał **1460 przykładów** i 79 cech opisujących domy, takie jak:
  + Numeryczne: GrLivArea, LotArea, OverallQual.
  + Kategoryczne: Neighborhood, HouseStyle, Exterior1st.
* **Zbiór testowy** liczył **1459 przykładów** (bez etykiety SalePrice).
* Dane zawierały:
  + **Braki danych**: Niektóre cechy miały brakujące wartości (np. LotFrontage, GarageYrBlt).
  + **Outliery**: Obserwacje odstające w cechach takich jak GrLivArea.

**Wstępne przetwarzanie danych:**

1. Braki danych:
   * Numeryczne wartości uzupełniono medianą.
   * Kategoryczne wartości uzupełniono trybem.
2. Transformacje:
   * Log-transformacja zmiennych z prawoskośnym rozkładem (SalePrice, GrLivArea).
   * Skalowanie cech numerycznych do zakresu [0, 1].
3. Kodowanie zmiennych kategorycznych:
   * One-Hot Encoding dla cech takich jak Neighborhood.

# Modele

W projekcie porównano działanie 4 modeli:

1. **Regresja liniowa bez regularyzacji**:
   * **Cel**: Punkt odniesienia dla bardziej zaawansowanych modeli.
   * **Implementacja**: W scikit-learn z LinearRegression.
2. **Regresja liniowa z regularyzacją (L2)**:
   * **Cel**: Zapobieganie nadmiernemu dopasowaniu przy wielu cechach.
   * **Implementacja**: W scikit-learn z Ridge.
   * **Parametry**: Optymalizacja parametru alpha metodą Grid Search.
3. **Gradient Boosting (XGBoost)**:
   * **Cel**: Wykorzystanie modeli drzew decyzyjnych do zwiększenia dokładności.
   * **Implementacja**: W XGBoost.
   * **Parametry**: Liczba estymatorów = 100, learning rate = 0.1.
4. **Sieć neuronowa (MLP)**:
   * **Architektura**: 2 warstwy ukryte (128 i 64 neurony) z funkcją aktywacji ReLU.
   * **Optymalizator**: Adam.
   * **Implementacja**: W TensorFlow/Keras.
   * **Regularizacja**: Dropout na poziomie 0.2 w warstwach ukrytych.

# Ewaluacja

Do ewaluacji wykorzystano metryki regresji:

* **Root Mean Squared Error (RMSE)**.
* **Mean Absolute Error (MAE)**.

|  |  |  |
| --- | --- | --- |
| **Model** | **RMSE** | **MAE** |
| Regresja liniowa bez regularyzacji |  |  |
| Regresja liniowa z regularyzacją |  |  |
| Gradient Boosting (XGBoost) |  |  |
| Sieć neuronowa (MLP) |  |  |

# Wnioski

1. **Sieć neuronowa (MLP)** osiągnęła najlepsze wyniki pod względem RMSE i MAE, co potwierdza zdolność do modelowania złożonych relacji w danych.
2. **Gradient Boosting (XGBoost)** niewiele ustępował sieci neuronowej, oferując bardziej interpretowalne wyniki dzięki analizie ważności cech.
3. **Regresja liniowa z regularyzacją (L2)** osiągnęła lepsze wyniki niż wersja bez regularyzacji, co wskazuje na znaczenie kontrolowania złożoności modelu.
4. **Regresja liniowa bez regularyzacji** była najprostsza w implementacji, ale miała najgorsze wyniki.