Szymon Jóźwiak, Uczenie Maszynowe 2024/2025

Cel projektu

Celem projektu było stworzenie modeli, które przewidują cenę pojazdu (dla modeli regresyjnych) lub kategorię ceny (`low`, `medium`, `high`) dla modeli klasyfikacyjnych. W analizie uwzględniono różnorodne podejścia, w tym modele regresji i klasyfikacji, aby porównać ich skuteczność w przewidywaniu cen na podstawie cech takich jak producent, model, rok produkcji, region sprzedaży i stan techniczny pojazdu.

Dane

Dane pochodzą z zestawu **„Craigslist Used Cars Dataset”**, dostępnego na platformie Kaggle:
https://www.kaggle.com/datasets/austinreese/craigslist-carstrucks-data?resource=download

Podział danych:

- **Łączna liczba przykładów**: 45,246.
- **Zbiór uczący**: 36,198 przykładów (80% danych).
- **Zbiór testowy**: 9,048 przykładów (20% danych).

Przed przetwarzaniem odrzucono niepoprawne dane, np. ceny mniejsze niż 500 USD lub większe niż 100,000 USD. Cechy kategoryczne zostały zakodowane za pomocą `LabelEncoder`, a cechy numeryczne zostały znormalizowane.

Dane zostały dodatkowo przygotowane do użytku w przyszłych analizach i wdrożeniach poprzez zapisanie modelu i obiektów przetwarzania w formacie .pkl. Wytrenowany model, skaler oraz encodery cech kategorycznych zostały zapisane, co umożliwia szybkie ładowanie i wykorzystanie w przyszłości bez konieczności ponownego trenowania lub przetwarzania danych. Takie podejście gwarantuje spójność przetwarzania oraz oszczędza czas w kontekście wdrożeń produkcyjnych.

Modele

Modele regresyjne

1. **Random Forest Regression**:
    - Model oparty na 100 drzewach decyzyjnych.
    - Wykorzystano wartości cech numerycznych i kategorycznych do przewidywania ceny pojazdu.
    - Ewaluacja: Mean Squared Error (MSE) i R-squared (R^2).
2. **Regresja wielomianowa (5. stopnia)**:
    - Model regresji wielomianowej przekształcający cechy do 5. stopnia (daje on lepsze wyniki od niższych stopni).
    - Ewaluacja: Mean Squared Error (MSE) i R-squared (R^2).
    - Dane zostały znormalizowane przed użyciem.

Modele klasyfikacyjne

3. **Regresja logistyczna z regularyzacją (L2)**:
    - Zastosowano solver `lbfgs` i regularyzację L2.
    - Dane znormalizowano, a klasy zbalansowano za pomocą `class_weight='balanced'`.
4. **Regresja logistyczna bez regularyzacji**:
    - Model regresji logistycznej bez użycia regularyzacji (`penalty=None`).
    - Zastosowano solver `lbfgs`.
    - **Zbalansowanie klas:** Włączono balansowanie klas za pomocą parametru class_weight='balanced', aby model nie faworyzował klas z większą liczbą przykładów.
5. **Naiwny klasyfikator Bayesowski**:
    - Gaussian Naive Bayes użyty do klasyfikacji danych.
    - Cechy kategoryczne (`manufacturer`, `model`, `region`, `condition`) zostały zakodowane liczbowo za pomocą `LabelEncoder`.
    - Model jest bardzo szybki w treningu i predykcji, ale może mieć ograniczenia w przypadku skorelowanych cech (np. `manufacturer` i `model`).
6. **K-Nearest Neighbors (KNN)**:
    - Algorytm KNN został zastosowany z liczbą sąsiadów ustawioną na **5** (`n_neighbors=5`).
    - Dane zostały znormalizowane za pomocą `StandardScaler`, co jest kluczowe dla algorytmu KNN, ponieważ opiera się on na odległościach między punktami.
    - Użyto domyślnego ważenia sąsiadów (`weights='uniform'`), co oznacza, że wszyscy sąsiedzi mieli równy wpływ na predykcję.
    - Liczba sąsiadów (`n_neighbors`) nie została zoptymalizowana za pomocą walidacji krzyżowej. Można by poprawić wyniki poprzez zastosowanie `GridSearchCV` w celu znalezienia optymalnej liczby sąsiadów.
    - Metryka odległości Euklidesowa (domyślnie w sklearn.neighbors)

Ewaluacja

Dla modeli regresyjnych zastosowano metryki **Mean Squared Error (MSE)** i **R-squared (R^2)**, a dla klasyfikacyjnych metryki **accuracy**, **precision**, **recall** oraz **F1-score**.

Wyniki modeli regresyjnych:

Model	MSE	R^2
Random Forest Regression	22,324,024	0.874
Regresja wielomianowa (5)	96,176,839	0.457

Wyniki modeli klasyfikacyjnych:

Model	Accuracy	Precision	Recall	F1-score
Regresja logistyczna z regularyzacją	0.63757	0.64	0.64	0.64
Regresja logistyczna bez regularyzacji	0.63755	0.64	0.64	0.64
Naiwny klasyfikator Bayesowski	0.61538	0.59	0.62	0.59
K-Nearest Neighbors (KNN)	0.75534	0.75	0.76	0.75

Wnioski

Modele regresyjne:

1. **Random Forest Regression**:
    - Osiągnął najlepsze wyniki wśród modeli regresyjnych:
        - **$R^2 = 0.874$** wskazuje, że model wyjaśnia 87% zmienności danych.
        - Najniższy **MSE** (22,324,024) oznacza, że błędy przewidywań są stosunkowo małe.
    - Model ten radzi sobie bardzo dobrze z danymi o nieliniowych relacjach między cechami.

2. **Regresja wielomianowa (5. stopnia)**:
    - Miała znacznie gorsze wyniki:
        - **R^2 = 0.457** sugeruje, że model wyjaśnia jedynie 46% zmienności danych.
        - Wysoki **MSE** (96,176,839) wskazuje na dużą rozbieżność między przewidywaniami a rzeczywistymi wartościami.
    - Wyniki te mogą wynikać z nadmiernego dopasowania do danych treningowych i braku zdolności modelu do uogólnienia.

Modele klasyfikacyjne:

1. **Regresja logistyczna z regularyzacją (L2)**:
    - Osiągnęła **accuracy = 0.63757**, co czyni ją porównywalną z regresją bez regularyzacji.
    - **Precision, Recall, i F1-score = 0.64** dla wszystkich klas sugerują, że model radzi sobie równomiernie dobrze na każdej kategorii.
    - Regularyzacja L2 zapobiega nadmiernemu dopasowaniu, co czyni model bardziej stabilnym.
2. **Regresja logistyczna bez regularyzacji**:
    - Wyniki były bardzo zbliżone do regresji z regularyzacją:
        - **accuracy = 0.63755**, **Precision, Recall, i F1-score = 0.64**.
    - Brak regularyzacji może prowadzić do nadmiernego dopasowania w większych zestawach danych, ale tutaj nie wpłynęło to znacząco na wyniki.
3. **K-Nearest Neighbors (KNN)**:
    - Uzyskał najlepsze wyniki wśród modeli klasyfikacyjnych:
        - **accuracy = 0.75534**, **Precision = 0.75**, **Recall = 0.76**, **F1-score = 0.75**.
    - KNN lepiej radzi sobie z danymi o złożonych, nieliniowych zależnościach, co tłumaczy lepsze wyniki.
    - Model może być zoptymalizowany poprzez dalsze strojenie hiperparametrów (np. liczby sąsiadów, metryki odległości), natomiast osiągnięto satysfakcjonujący wynik.
4. **Naiwny klasyfikator Bayesowski**:
    - Uzyskał najniższe wyniki:
        - **accuracy = 0.61538**, **Precision = 0.59**, **Recall = 0.62**, **F1-score = 0.59**.
    - Słabe wyniki wynikają z założenia niezależności cech, które nie jest spełnione w danych (np. `manufacturer` i `model` są silnie skorelowane).

```
🤠🤠🤠 → project python3 random_forrest_regression.py
Mean Squared Error: 22324024.56839664
R-squared: 0.8740903533947382
Sample of predictions:
       manufacturer                    model    year                  region  condition  actual_price  predicted_price
385285         ford                    f-150  2017.0           wichita falls   like new         25988     25812.406667
169884         ford      super duty f-550 drw  2011.0                 wichita       good         26990     33620.136667
50199           ram                1500 4x4  2014.0             reno / tahoe  excellent         15000     23445.560173
59418           gmc  sierra 1500 crew cab sle  2014.0            santa barbara  excellent         25983     26395.801429
394635       nissan           350z enthusiast  2008.0  norfolk / hampton roads       good          9000     14542.230000
231262          bmw              x5 xdrive50i  2012.0                    boone  excellent         11990     11991.663391
346099        lexus  nx 200t sport utility 4d  2015.0                 columbia       good         22590     24113.420000
65179           ford                    escape  2019.0                 stockton  excellent         19999     20298.530000
293959          ford                      f750  2019.0               cleveland        new         97500     51714.760000
221458         honda                     civic  2010.0              springfield       good          5500      6967.700000
🤠🤠🤠 → project python3 regresja_wielomianowa.py
Mean Squared Error: 103246554.63066114
R-squared: 0.41767949739848065
```

```
🤠🤠🤠 → project python3 regresja_logistyczna_z_reg.py
Accuracy: 0.6375751414427157
Classification Report:
              precision    recall  f1-score   support

        high       0.69      0.79      0.74     17212
         low       0.75      0.65      0.69     16329
      medium       0.41      0.40      0.41     11707

    accuracy                           0.64     45248
   macro avg       0.62      0.61      0.61     45248
weighted avg       0.64      0.64      0.64     45248

🤠🤠🤠 → project python3 regresja_logistyczna_bez_reg.py
Accuracy: 0.6375530410183875
Classification Report:
              precision    recall  f1-score   support

        high       0.69      0.79      0.74     17212
         low       0.75      0.65      0.69     16329
      medium       0.41      0.40      0.41     11707

    accuracy                           0.64     45248
   macro avg       0.62      0.61      0.61     45248
weighted avg       0.64      0.64      0.64     45248
```

```
🤠🤠🤠 → project python3 naiwny_bayes.py
Accuracy: 0.615386315417256
Classification Report:
              precision    recall  f1-score   support

        high       0.61      0.78      0.68     17212
         low       0.69      0.73      0.71     16329
      medium       0.42      0.21      0.28     11707

    accuracy                           0.62     45248
   macro avg       0.57      0.57      0.56     45248
weighted avg       0.59      0.62      0.59     45248

🤠🤠🤠 → project python3 knn.py
Accuracy: 0.7553483026874116
Classification Report:
              precision    recall  f1-score   support

        high       0.81      0.86      0.83     17212
         low       0.78      0.80      0.79     16329
      medium       0.62      0.54      0.57     11707

    accuracy                           0.76     45248
   macro avg       0.74      0.73      0.73     45248
weighted avg       0.75      0.76      0.75     45248
```