Raport: Pierwsza model uczenia maszynowego

W ramach danej części projektu dla analizy danych został wybrany dataset dotyczący kosztów edukacji międzynarodowej. Ten dataset został stworzony w celu analizy i porównania kosztów studiowania za granicą. Obejmuje ponad 900 programów z uniwersytetów na całym świecie, w tym dane dotyczące utrzymania w kraju, zakwaterowania, opłat wizowych i ubezpieczenia. Dataset jest złożony z 907 wierszy i 12 column oraz zawiera 5 cech kategorialnych (kraj, miasto, uniwersytet, program i poziom edukacji) i 7 cech numerycznych (czas trwania, koszty utrzymania, czynsz, opłata wizowa, ubezpieczenie, kurs wymiany i całkowitą kwotę)

```
rows, columns: (907, 12)
```

Podczas analizy omawianego wyżej datasetu stwierdziłam, że będę rozwiązywała problem regresji i spróbowałam przewidzieć jaki będzie koszt całkowity studiowania w danym kraju na danej uczelni. Czyli moją zmienną docelową stała się zmienna Tuition_USD.

1. Porównanie trzech różnych modeli uczenia maszynowego: regresja liniowa, drzewo decyzyjne i svm.

Model	Zbiór	Plik .csv	MAE	RMSE
Regresja liniowa	train	ln_train_predictions.csv	1519.98	2535.63
Regresja liniowa	validate	ln_validate_predictions.csv	346934.95	850400.51
Regresja liniowa	test	ln_test_predictions.csv	1119841.62	3588598.48
Drzewo decyzyjne	train	dd_train_predictions.csv	0.0	0.0
Drzewo decyzyjne	validate	dd_validate_predictions.csv	3046.7	4435.76
Drzewo decyzyjne	test	dd_test_predictions.csv	2679.12	3954.45
SVM	train	svr_train_predictions.csv	14215.35	16846.11
SVM	validate	svr_validate_predictions.csv	21284.31	22149.94
SVM	test	svr_test_predictions.csv	21353.98	21792.01

```
Model:  ln , set:  train
Mean average error:  1519.98
Rout mean squared error:  2535.63

Model:  ln , set:  validate
Mean average error:  346934.95
Rout mean squared error:  850400.51

Model:  ln , set:  test
Mean average error:  1119841.62
Rout mean squared error:  3588598.48

Model:  dd , set:  train
Mean average error:  0.0
Rout mean squared error:  0.0

Model:  dd , set:  validate
Mean average error:  3046.7
Rout mean squared error:  4435.76

Model:  dd , set:  test
Mean average error:  2679.12
Rout mean squared error:  3954.45

Model:  svr , set:  train
Mean average error:  14215.35
Rout mean squared error:  16846.11

Model:  svr , set:  validate
Mean average error:  21284.31
Rout mean squared error:  22149.94

Model:  svr , set:  test
Mean average error:  21353.98
Rout mean squared error:  21792.01
```

Wynik: regresja liniowa jest mocno przekwalifikowana, to wynika z wielkich błędów w walidacji i teście. Drzewo decyzyjne pokazuje najlepszy wynik, podaj najlepszyj wynik na zbiorze treningowym. SVM jest najstabilniejszą metodą, ale z wielkimi błędami. Optymalnym wyborem jest drzewo decyzyjne.

2. Samodzielna implementacja
    1. Regresja liniowa
        a. Z zamkniętą formułą($\theta = (X^T X)^{-1} X^T y$)

Model	Zbiór	Plik .csv	MAE	RMSE
Regresja liniowa	train	ln_train_predictions.csv	1519.98	2535.63

Regresja liniowa	validate	ln_validate_predictions.csv	346934.95	850400.51
Regresja liniowa	test	ln_test_predictions.csv	1119841.62	3588598.48
Samodzielna implementacja regresji liniowej	train	my_ln_train_predictions.csv	909.19	1901.21
Samodzielna implementacja regresji liniowej	validate	my_ln_val_predictions.csv	$1.46 * 10^{35}$	$2.22 * 10^{35}$
Samodzielna implementacja regresji liniowej	test	my_ln_test_predictions.csv	$1.64*10^{46}$	$2.17 * 10^{46}$

```
Model: my ln, set:  train
Mean average error:  909.19
Rout mean squared error:  1901.21

Model: my ln, set:  val
Mean average error:  1.4558756404246343e+35
Rout mean squared error:  2.2234722050798 47e+35

Model: my ln, set:  test
Mean average error:  1.6445905491667 78e+46
Rout mean squared error:  2.1679136754520899e+46
```

Wynik: Samodzielny model jest dobry w przetwarzaniu danych treningowych, ale daje katastrofalne błędy w walidacji i testach, co może wskazywać na wyciek danych.

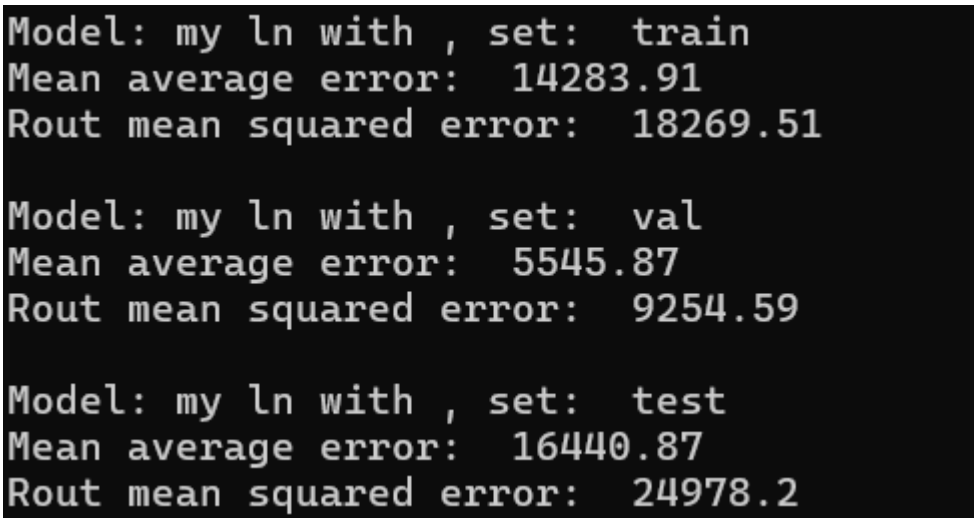
b. Ograniczenia

- Obliczanie macierzy odwrotnej $(X^T X)^{-1}$ staje się niedokładne w przypadku dużych wymiarów danych. Stád wynikają duże błędy w danych testowych i walidacyjnych
- Duża złożoność O(n^3). Ze względu na to program długo pracuje i potrzebuje dużo pamięci, co jest niepraktyczne.
- Metoda nie jest elastyczna

2. Regresja liniowa z gradientem

Model	Zbiór	Plik .csv	MAE	RMSE
Regresja liniowa	train	ln_train_predictions.csv	1519.98	2535.63

Regresja liniowa	validate	ln_validate_predictions.csv	346934.95	850400.51
Regresja liniowa	test	ln_test_predictions.csv	1119841.62	3588598.48
Samodzielna implementacja regresji liniowej z gradientem	train	my_ln_grad_predictions.csv	14283.91	18269.51
Samodzielna implementacja regresji liniowej z gradientem	validate	my_ln_grad_validate_predictions.csv	5545.87	9254.59
Samodzielna implementacja regresji liniowej z gradientem	test	my_ln_test_predictions.csv	16440.87	24978.2



Wynik: Implementacja z gradientem daje bardziej stabilne wyniki na wszystkich próbkach, bez katastrofalnego wzrostu błędów. Chociaż dokładność jest nadal niższa niż na danych treningowych, metoda jest odporna na przetrenowanie.

Wynik końcowy:
W eksperymencie porównano klasyczną regresję liniową, jej wersję z gradientem prostym, drzewo decyzyjne oraz SVM. Regresja liniowa była mocno przetrenowana – dobre wyniki na treningu, ale duże błędy na walidacji i teście. Zastosowanie gradientu dało stabilniejsze wyniki na wszystkich zbiorach, choć mniej dokładne na treningowym. Drzewo decyzyjne osiągnęło najlepsze wyniki ogólne i okazało się najbardziej dopasowane do danych. SVM był stabilny, ale generował zbyt duże błędy. Najlepszym wyborem okazało się drzewo decyzyjne, a regresja z gradientem może być dobrą opcją przy ryzyku przeuczenia.