Katzper Michno, 5.04.2022, MPUM

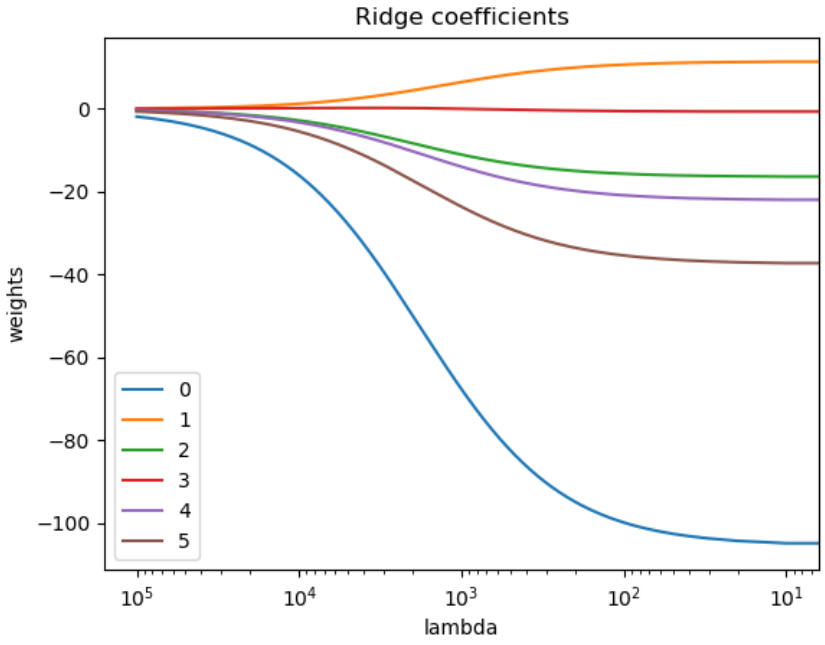
Miniprojekt 1

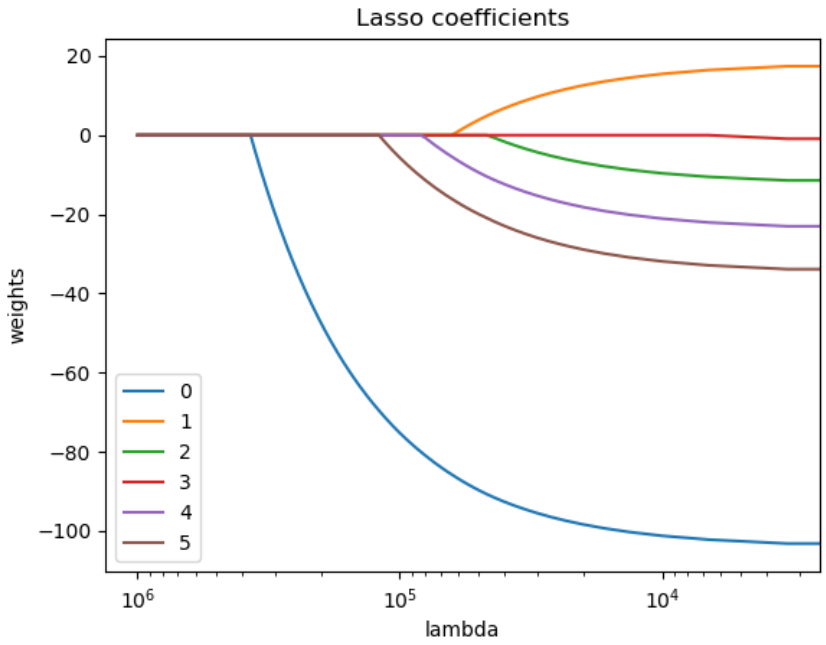
Celem projektu było zaprojektowanie i implementacja algorytmu regresji liniowej. Konkretnie, mieliśmy do czynienia z sześcioma cechami będącymi liczbami rzeczywistymi oraz jedną zmienną objaśnianą, także liczbą rzeczywistą. Zbiór danych z którego korzystaliśmy znajduje się w pliku *example.data*. W pliku *regression.py* znajduje się gotowy skrypt przeprowadzający regresję zaimplementowany w języku Python z wykorzystaniem biblioteki NumPy do przeprowadzania obliczeń. Wszystkie wykresy pojawiające się w raporcie zostały wygenerowane z użyciem biblioteki Matplotlib.

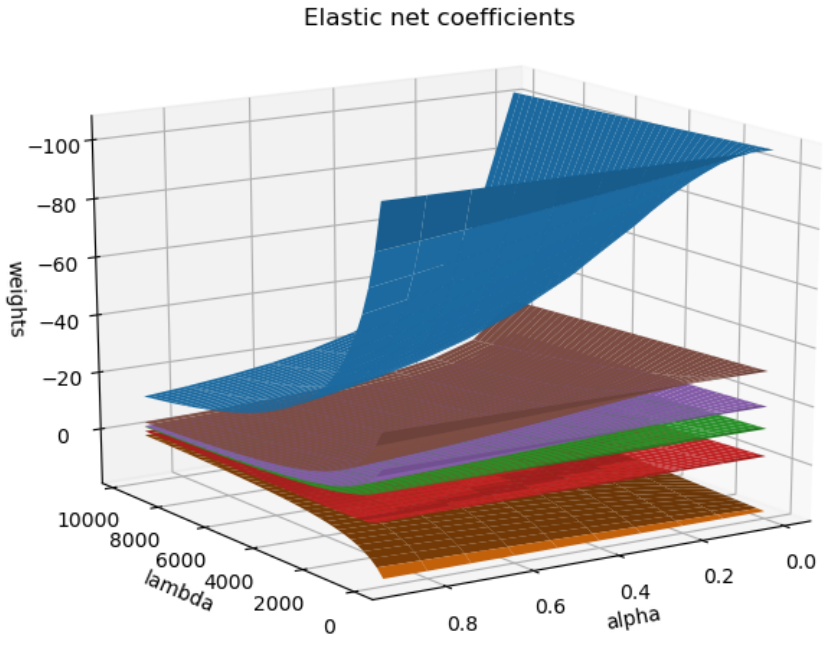
Wstępna analiza danych

Pierwszym krokiem była implementacja algorytmu regresji grzbietowej, lasso oraz z siecią elastyczną. W każdym przypadku korzystamy z kwadratowej funkcji straty. Rozwiązanie w problemie regresji grzbietowej znajdujemy w sposób analityczny, w przypadku zaś regresji lasso i sieci elastycznej zaimplementowany został algorytm spadku po współrzędnych. Przed wykonaniem jakichkolwiek obliczeń, zawsze standaryzujemy macierz planowania.

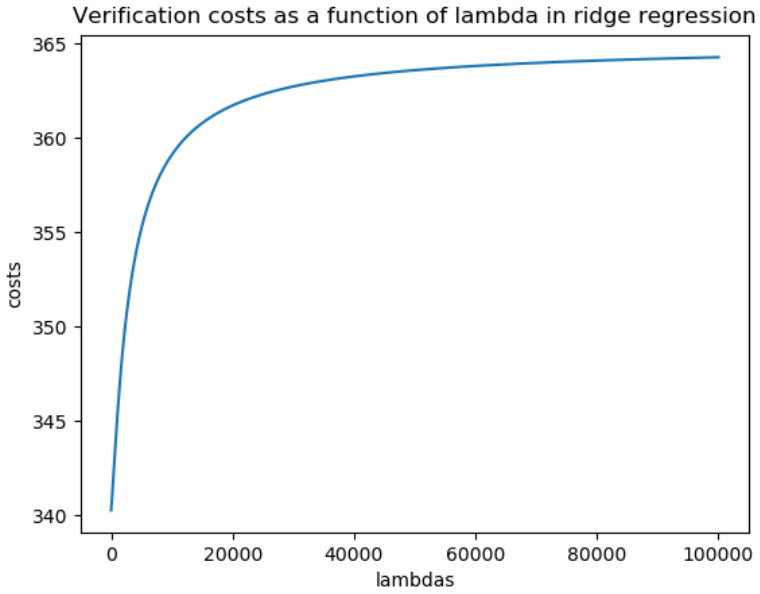
Po zastosowaniu regresji dla całego zbioru danych bez wykorzystania żadnych funkcji bazowych, otrzymaliśmy pierwsze rezultaty.

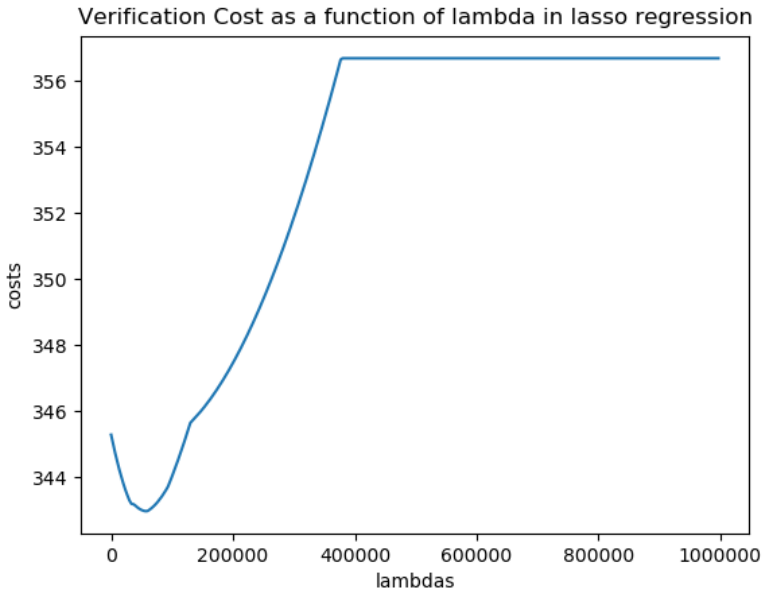



Lasso coefficients
20
0
−20
−40
−60
−80
−100
weights
0
1
2
3
4
5
10^6
10^5
10^4
lambda


Elastic net coefficients
−100
−80
−60
−40
−20
0
weights
10000
8000
6000
4000
2000
0
lambda
0.8
0.6
0.4
0.2
0.0
alpha

Po podziale zbioru danych na zbiór treningowy i weryfikacyjny, otrzymaliśmy także informację na temat błędu w zależności od parametrów regularyzacji.

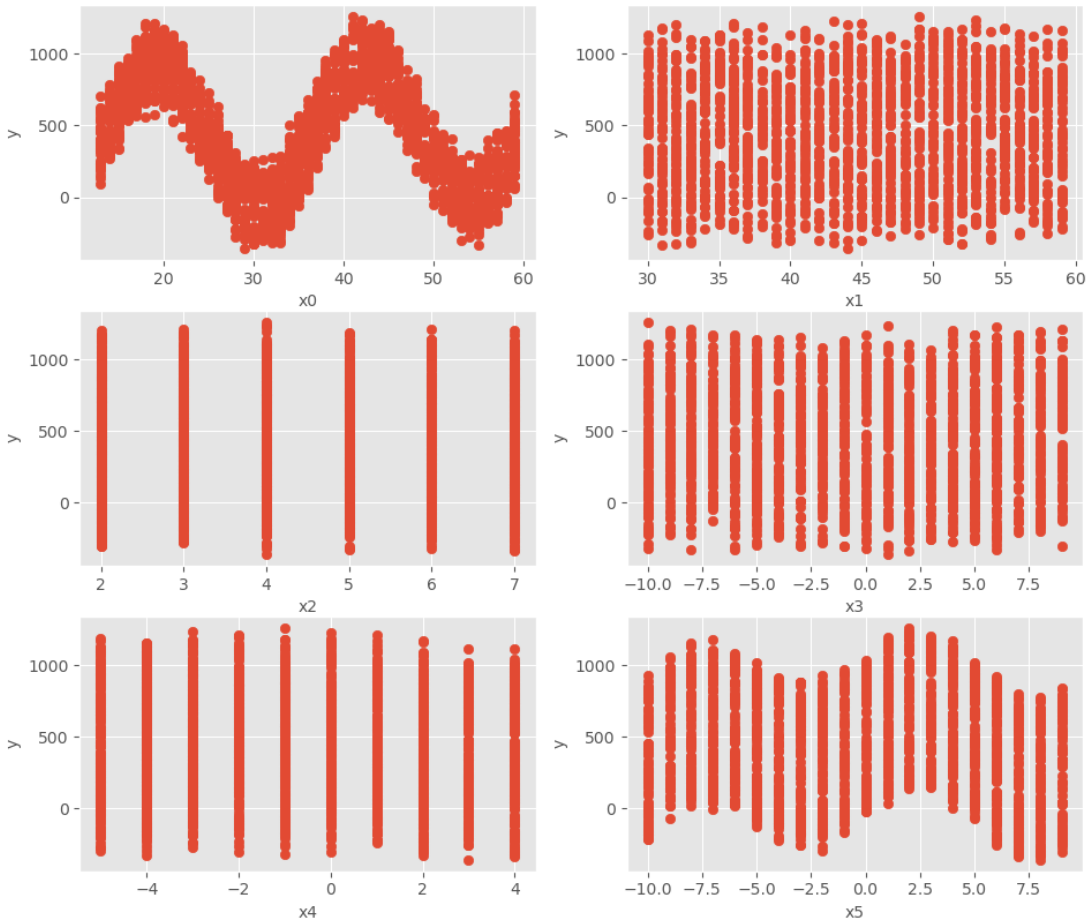




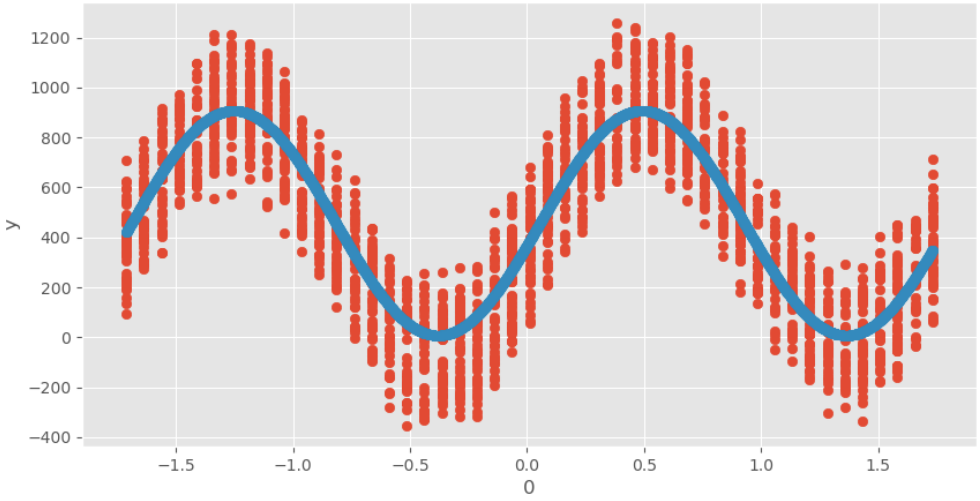
Błąd jest liczony jako spierwiastkowany średni błąd kwadratowy, czyli postaci $\sqrt{\frac{||X\theta - y||_2^2}{m}}$. Z powyższych wykresów jesteśmy w stanie wywnioskować dwie rzeczy. Po pierwsze, nie wszystkie cechy są równie istotne. Dzięki tej informacji będziemy w stanie ograniczyć się do mniejszej liczby potencjalnych modeli przy doborze funkcji bazowych. Po drugie, pomimo zastosowania różnych regularyzacji z szerokim zakresem wartości parametrów, średni błąd na zbiorze weryfikacyjnym nie spadł poniżej 300, z tego możemy wnioskować, że zmienna objaśniania nie jest w trywialny sposób zależna liniowo od cech i potrzebujemy bardziej skomplikowanych funkcji bazowych.

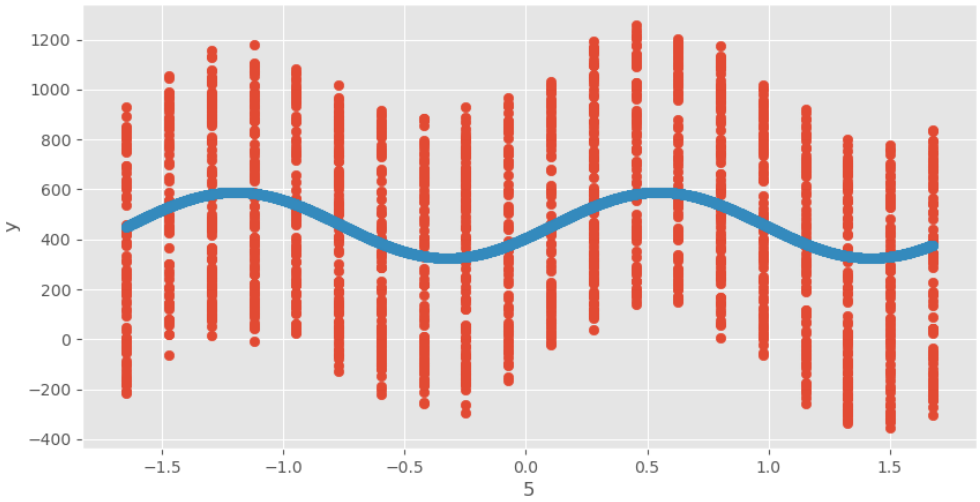
Szczegółowa analiza danych

Pomocnym okazało się wygenerowanie wykresów wartości objaśnianej od każdej cechy osobno.



Gołym okiem widać, że y zależy od x_0 i x_5 w sposób przypominający funkcję sin. Z pomocą algorytmu brutalnego udało się dopasować do nich funkcje $sin(3.6x - 0.2)$ i $sin(3.64x - 0.4)$ przemnożone przez pewne skalary, których znalezienie możemy pozostawić algorytmowi regresji liniowej. Podobnie udało się dopasować funkcję $sin(1.33x+1.76)$ do x_4, czyli kolejnej w kolejności istotności cechy. Nie poprawiło to jednak znacząco wyników regresji.





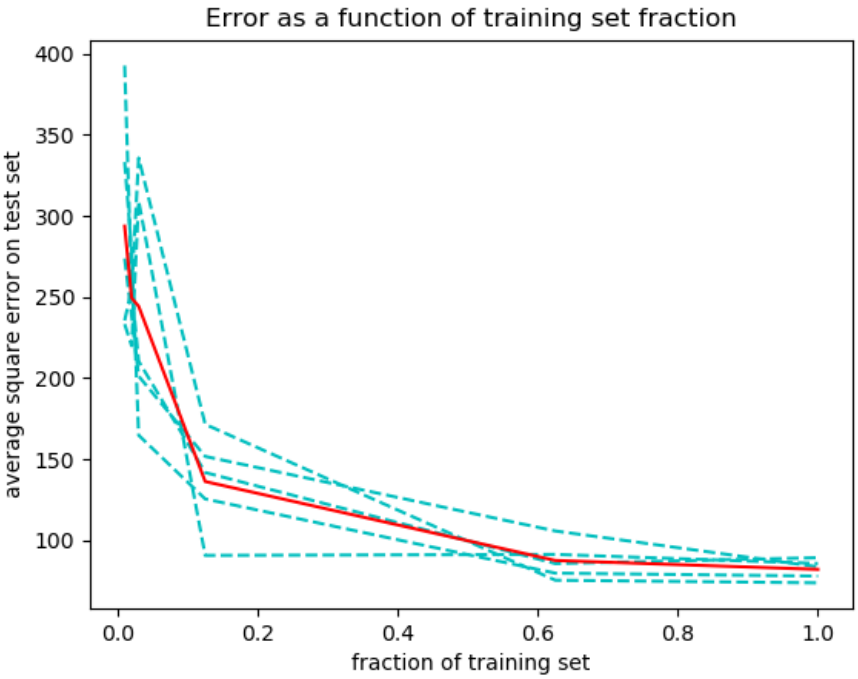
Funkcje bazowe

W formie eksperymentu zostało wypróbowanych wiele różnych funkcji bazowych: wielomianowych, gaussowskich i sinusów. Okazało się, że większość z nich nie przyniosła znaczących efektów, najlepsze okazały się sinusy przytoczone w poprzedniej sekcji. Poniżej znajduje się zestawienie wykorzystanych zestawów funkcji bazowych wraz z ich średnim błędem na zbiorze testowym. Dane zostały w sposób losowy podzielone na zbiór treningowy, weryfikacyjny i testowy w proporcjach $6:2:2$. Wybór optymalnej regularyzacji i jej parametrów został przeprowadzony z wykorzystaniem zbioru weryfikacyjnego. W każdym przypadku parametry regularyzacji należały do zakresów: $\lambda \in [0, 10^4]$ dla regresji grzbietowej, $\lambda \in [0, 10^6]$ dla regresji lasso oraz $\lambda \in [0, 10^4], \alpha \in [0, 1]$ dla elastycznej sieci i były równomiernie próbkowane.

Opis	Postać	Błąd
identyczność	$x_i \to x_i$ dla $i = 0...5$	352.4
kwadrat	$x_i \to x_i^2$ dla $i = 0...5$	362.8
sześcian	$x_i \to x_i^3$ dla $i = 0...5$	338.1
suma iloczynów każdej pary	$(x_i, x_j) \to x_i x_j$ dla $i, j = 0...5$	362.8
gauss[0.2]	$x_i \to exp\left(-\frac{x_i^2}{0.2^2}\right)$ dla $i = 0...5$	366.6
gauss[1]	$x_i \to exp\left(-x_i^2\right)$ dla $i = 0...5$	365.9
gauss[3]	$x_i \to exp\left(-\frac{x_i^2}{3^2}\right)$ dla $i = 0...5$	363.2
sinus dla x_0	$x_0 \to \sin(3.6x_0 - 0.2)$	160.9
sinus dla x_0, x_5	$x_0 \to \sin(3.6x_0 - 0.2)$, $x_5 \to \sin(3.64x_5 - 0.4)$	85.7
sinus dla x_0, x_4, x_5	$x_0 \to \sin(3.6x_0 - 0.2)$, $x_4 \to \sin(1.33x_4 + 1.76)$, $x_5 \to \sin(3.64x_5 - 0.4)$	84.1

Pełen przebieg

Na końcu pięciokrotnie został uruchomiony pełen przebieg algorytmu. Za każdym razem skrypt sam wybierał najlepszy zestaw funkcji bazowych i regularyzację. Dane nadal podzielone były w proporcjach $6:2:2$, w każdym przebiegu losowane na nowo, jednak tym razem testowaliśmy błąd w zależności od wielkości zbioru treningowego. W tym celu badaliśmy błąd dla frakcji zbioru treningowego: $0.01, 0.02, 0.03, 0.125, 0.625, 1$. Poniżej znajduje się wykres przedstawiający wyniki tej analizy. Niebieskie łamane to poszczególne przebiegi zaś czerwona linia pokazuje średnią ze wszystkich przebiegów.


Error as a function of training set fraction
average square error on test set
400
350
300
250
200
150
100
0.0
0.2
0.4
0.6
0.8
1.0
fraction of training set