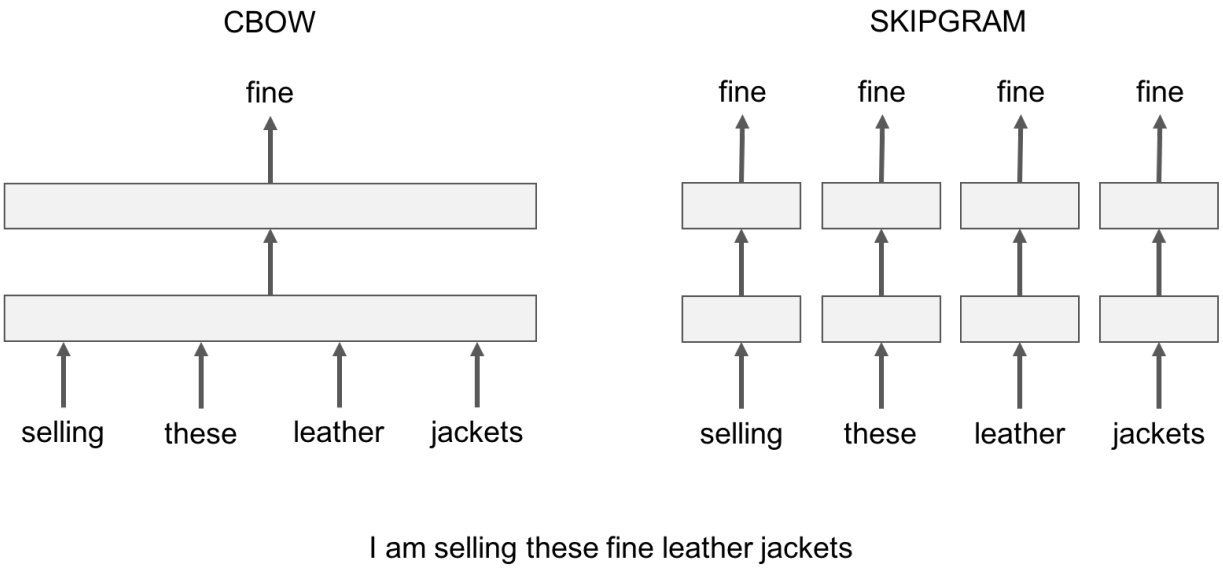
1 Modele dystrybucyjne

Oba użyte przeze mnie modele dystrybucyjne utworzyłem używajac biblioteki `fasttext`. Jeden z nich to model `cbow`, a drugi to `skipgram`. Oba modele przetwarzaja słowa w 100-wymiarowe wektory i używa rozmiaru okna kontekstowego 5.

`fasttext` bazuje na `word2vec`, a jego charakterystyka jest to że używa podczas uczenia informacje o znakach, z jakich składa sie słowo. Działanie opiera sie na dwuczęściowej sieci neuronowej składajacej sie z enkodera i dekodera. Nie jest to autokoder ponieważ jego celem nie jest odbudowywanie cech wejściowych. Dokładność takiego klasyfikatora ostatecznie nie jest ważna, ponieważ najwieksze znaczenie ma jakość tworzonych wektorów reprezentacji ukrytej słowa. Usprawnieniem odróżniajacym `fasttext` od `word2vec` jest wykorzystanie informacji nie tylko z kontekstu słowa, ale też ze znaków z jakich sie składa. Do reprezentacji słowa dodawane sa zahashowane n-gramy znaków z wewnatrz danego słowa. Domyślny przedział długości wych n-gramów to od 3 do 6 znaków. Pozwala to na uzyskanie lepszych reprezentacji dla słów rzadko wystepujacych w tekście, ponieważ o ile istnieje mało kontekstów dla takiego słow, model może przybliżyć jego reprezentacje do słów o podobnym zapisie.



1.1 `cbow`

W tym modelu wejściem sieci neuronowej sa słowa z kontekstu dla danego słowa, a wyjściem jest samo słowo. W ten sposów reprezentacja ukryta jest uzyskiwana przez zakodowywanie kontekstu, a nie samego słowa. Mimo to wyuczenie sieci wymaga, żeby dekoder był w stanie z reprezentacji ukrytej jednoznacznie odtworzyć dane słowo.

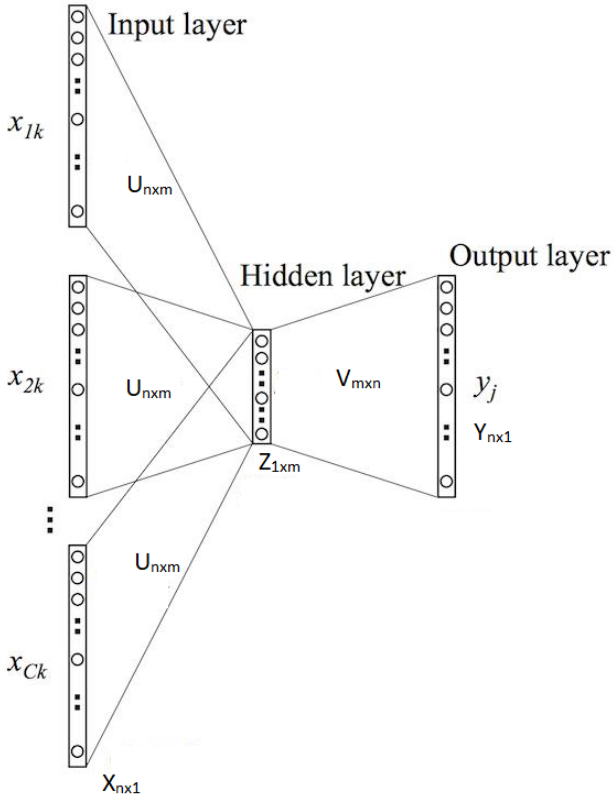


Figure 1: cbow

1.2 `skipgram`

`skipgram` działa odwrotnie do `cbow`. Zakodowuje on słowo do postaci ukrytej, a nastepnie odtwarza jego kontekst. Wymusza to podobna reprezentacje dla słów używanych w podobnych kontekstach.

2 Regresja

Do regresji użyłem własnej implementacji rekurencyjnej sieci neuronowej. Wybrałem taka architekture sieci ze wzgledu na zmienna długość danych wejściowych. Sieć składa sie z 3 warstw rekurencyjnych (z warstwami ReLU pomiedzy) i jednej warsty liniowej. Warsty rekurencyjne na wejściu dostaja wektory długości 100 i zwracaja wektor i stan ukryty też długości 100. Warstwa liniowa przyjmuje wektor długości 100 i zwraca pojedyncza liczbe bedaca przewidziana punktacja posta/komentarza na Reddicie. Sieć uczona była przez 50 epok. Poniżej sa wykresy miar MSE i R2 po każdej epoce. Model `m1` to `cbow`, a `m2` to `skipgram`.

Wykresy miary MSE sa pokazane osobno, ponieważ poczatkowy stan MSE jest mocno zależne od poczatkowych wartości parametrów regresora, które sa losowe. Duże różnice uniemożliwiaja bezpośrednie porównanie miary MSE w trakcie uczenie. Można tylko patrzeć na kształt wykresów.

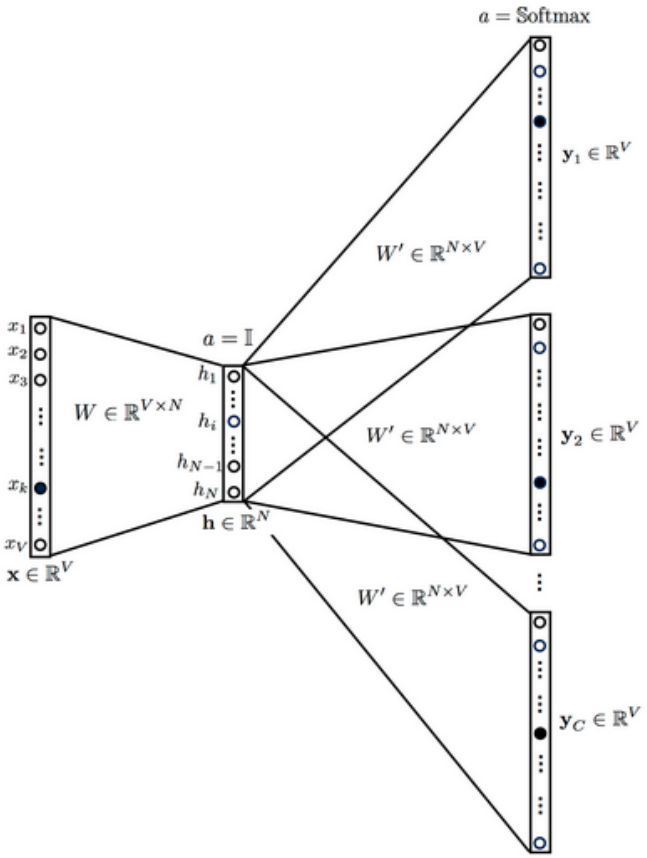


Figure 2: skipgram

Jak widać sieć ma duża skłonność do przeuczania sie. Wartości metryk dla zbioru walidacyjnego pogarszaja sie mniej wiecej w takim tepie, jak polepszaja sie dla zbioru treningowego. Dla zbioru treningowego znacznie lepiej radzi sobie model `cbow`. Oba modele daja lepsze MSE jeżeli sa wyuczone na korpusie wzorcowym. Oba zjawiska widać zarówno na wykresie R2, jak i wykresach MSE.

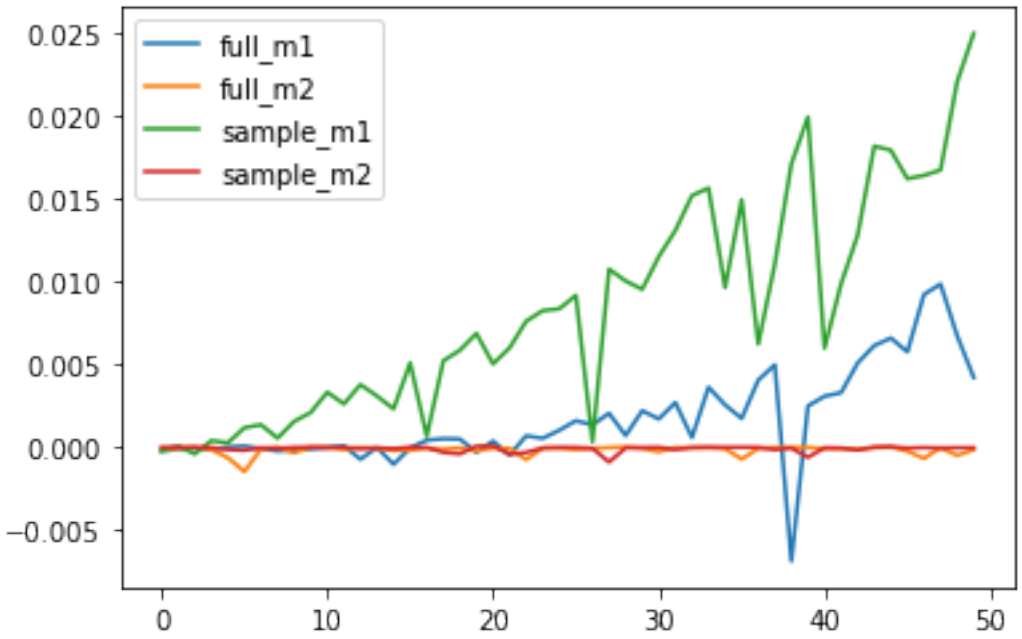


Figure 3: r2 score dla zbioru treningowego

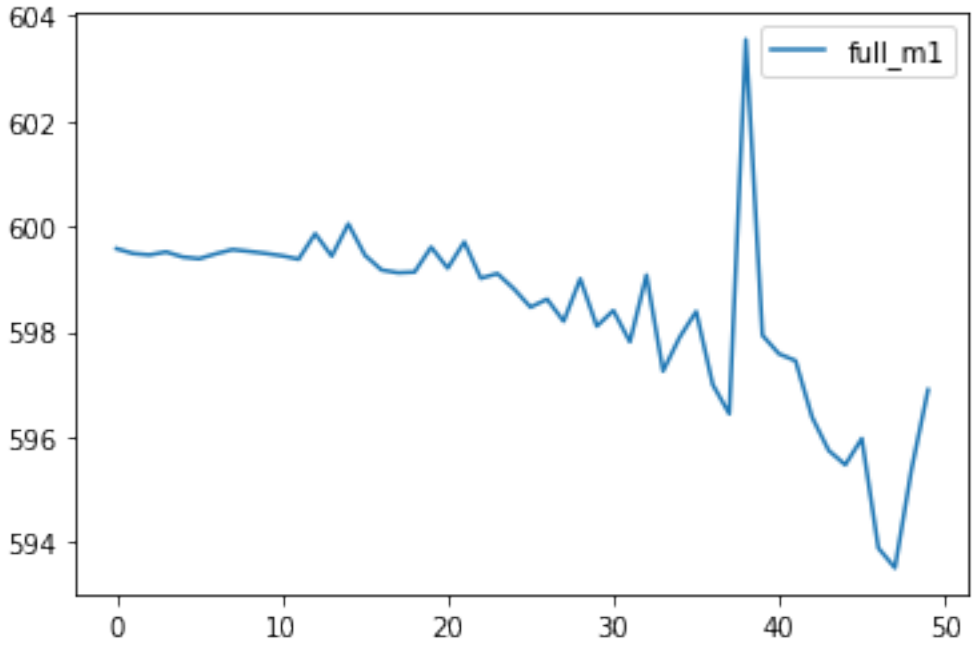


Figure 4: mse score dla zbioru treningowego

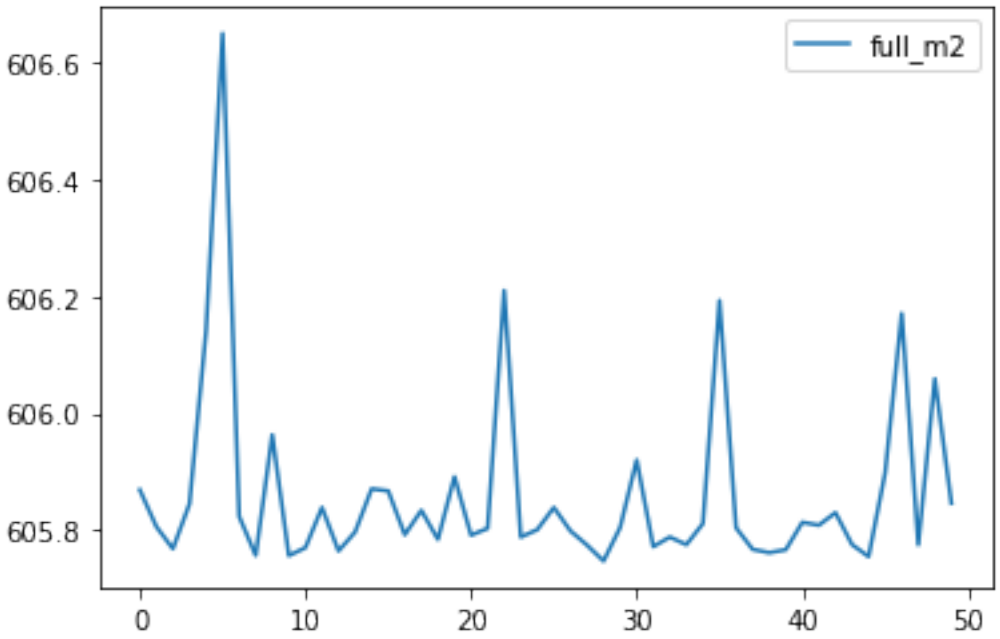


Figure 5: mse score dla zbioru treningowego

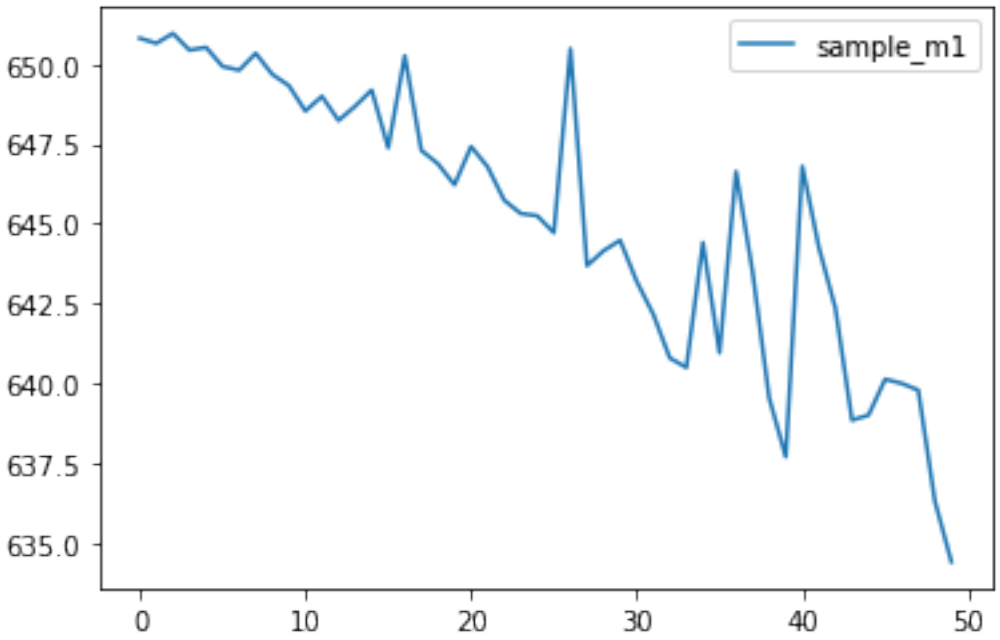


Figure 6: mse score dla zbioru treningowego

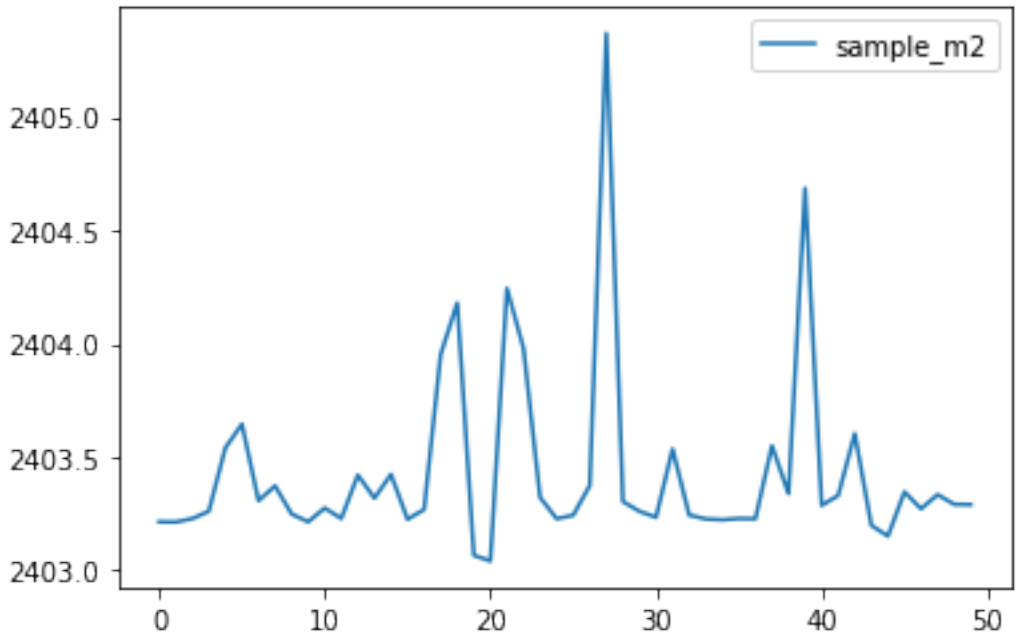


Figure 7: mse score dla zbioru treningowego

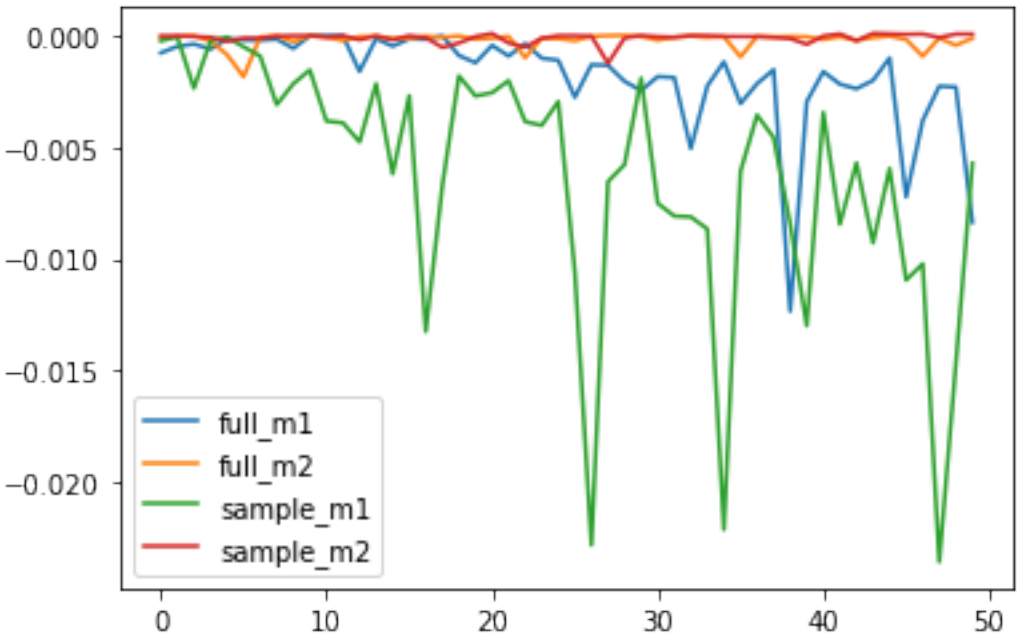


Figure 8: r2 score dla zbioru walidacyjnego

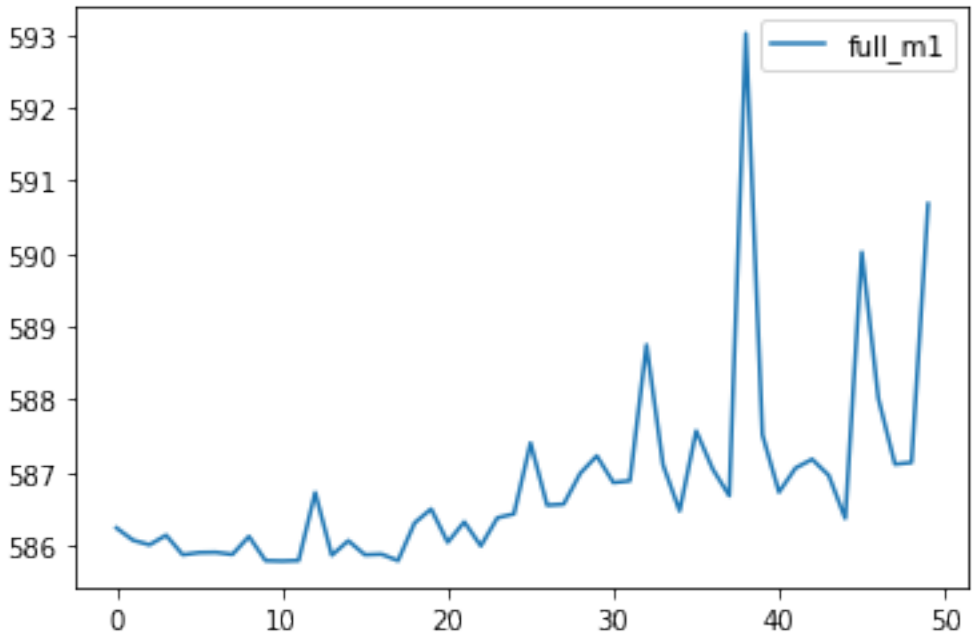


Figure 9: mse score dla zbioru walidacyjnego

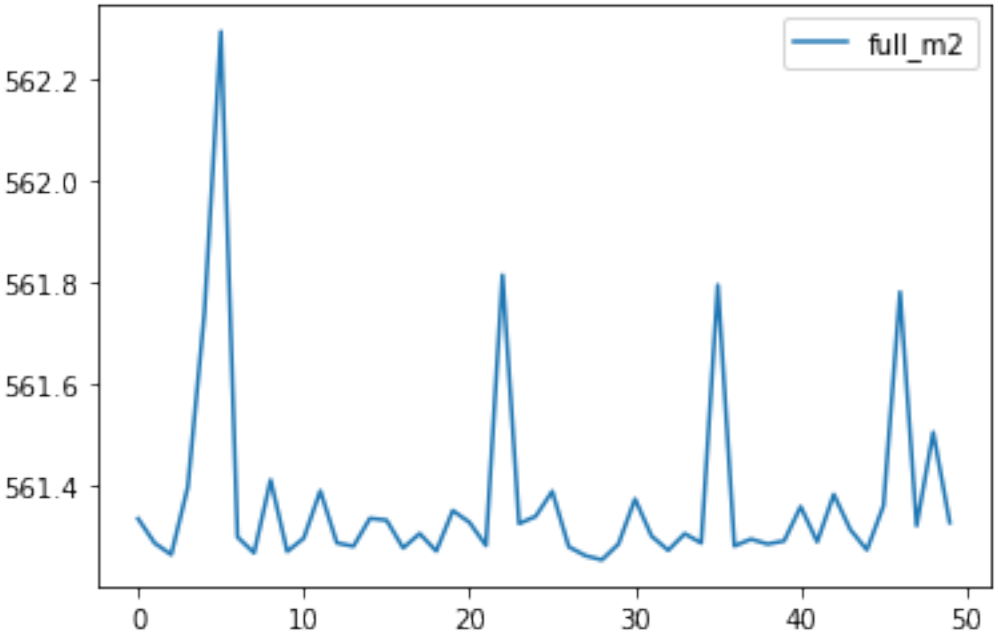


Figure 10: mse score dla zbioru walidacyjnego

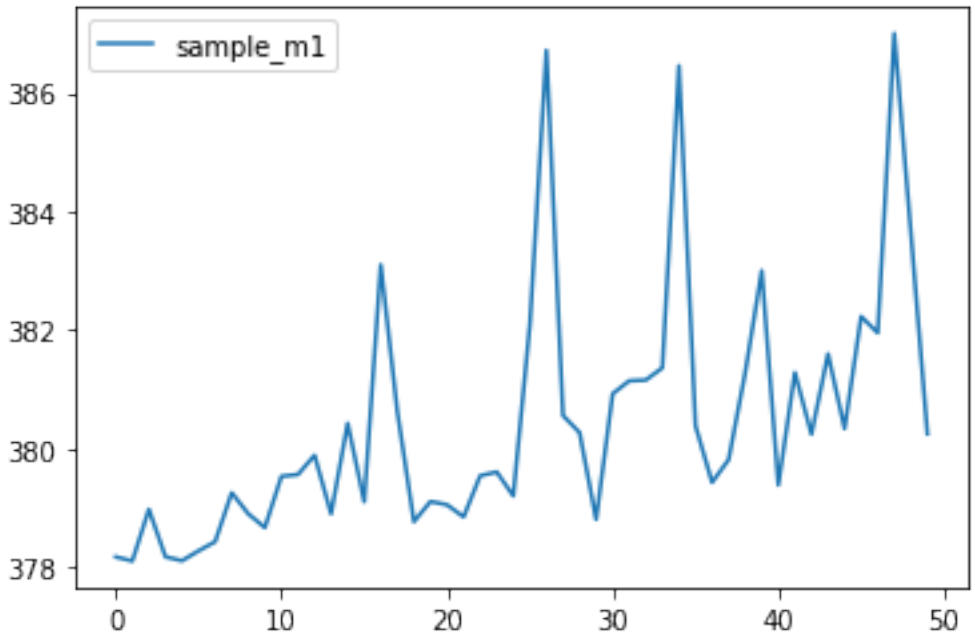


Figure 11: mse score dla zbioru walidacyjnego

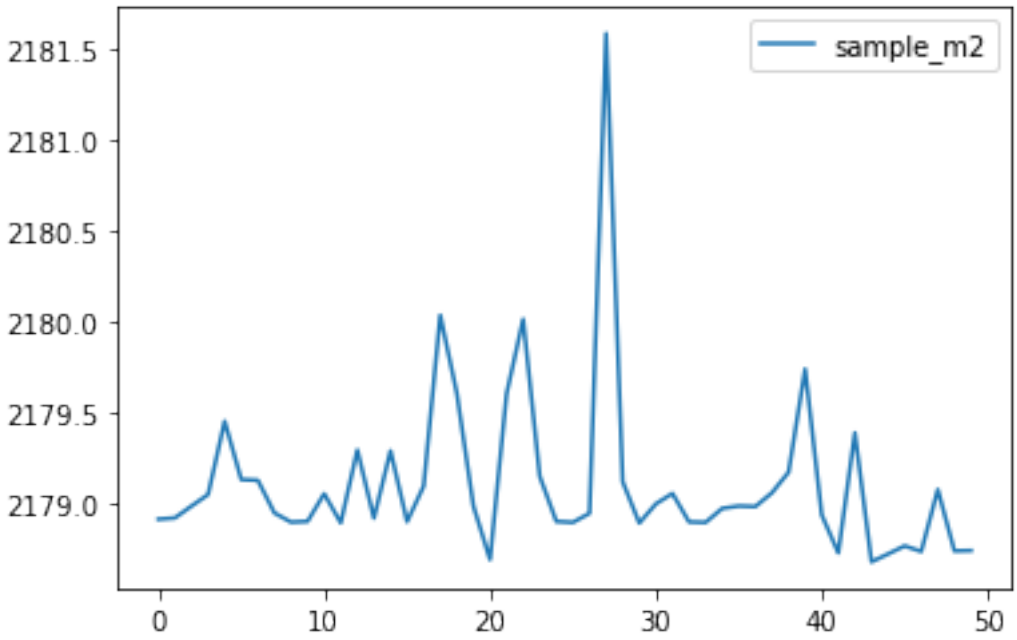


Figure 12: mse score dla zbioru walidacyjnego