Klasyfikacja na danych Climate Model Simulation

Maciej Złotorowicz

05.01.2023

Streszczenie

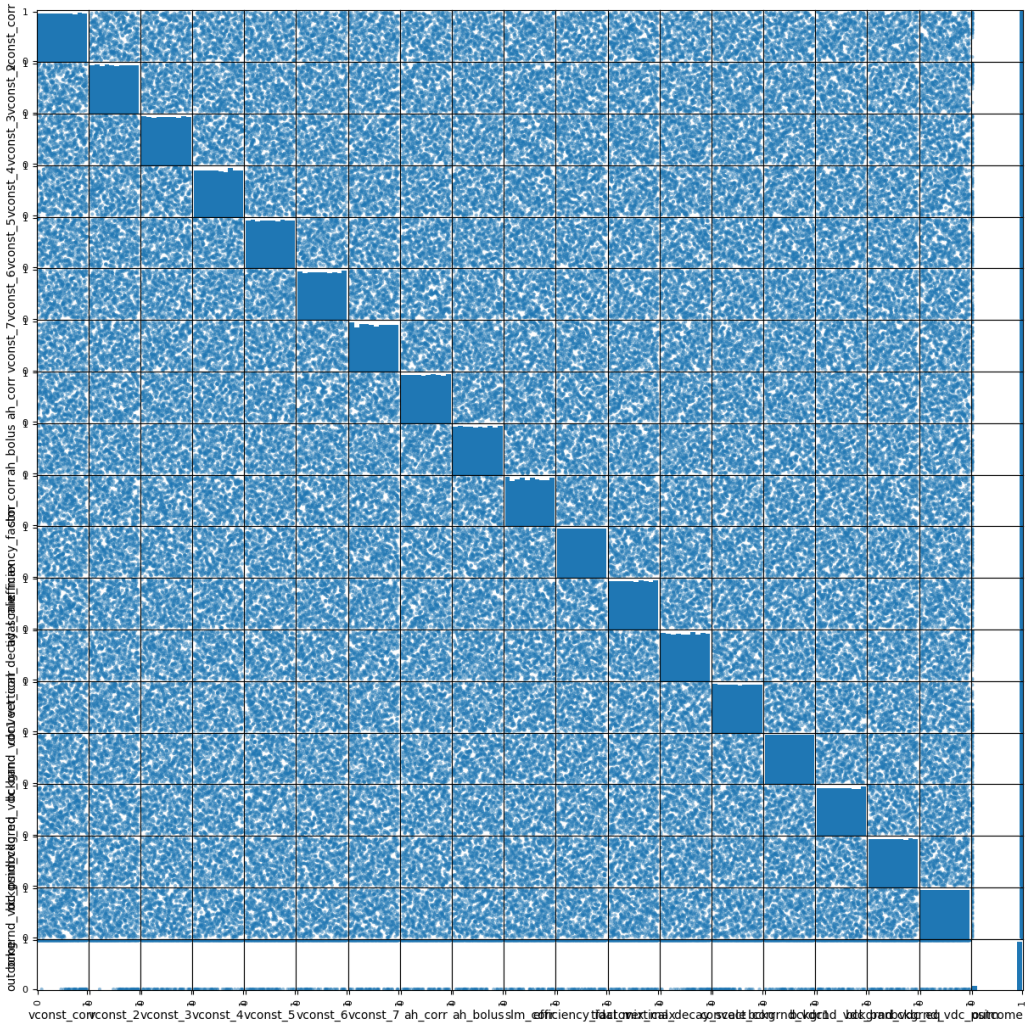
Projekt dotyczy zbioru danych `climate_model_sim_crashes` . Zawiera on **540** rekordów z **21** cechami a zadaniem jest klasyfikacja binarna.

Zbiór danych posiada kilka charakterystycznych cech - jest on bardzo niezbalansowany a cechy są od siebie całkowicie niezalezne. sprawia to że wiele algorytmów klasyfikacji nie radzi sobie z tym zbiorem danych a wiele technik - na przykład redukcji wymiarowości PCA czy trenowanie nienadzorowane się tutaj nieprzydadzą.

Został wybrany ten zbiór danych właśnie ze względu na te utrudnienia.

Zostały tutaj zastosowane podstawowe algorytmy klasyfikacji binarnej - regresja logistyczna, drzewa decyzyjne, maszyny wektorów nośnych sieci neuronowe ale z bardzo silną regularyzacją.

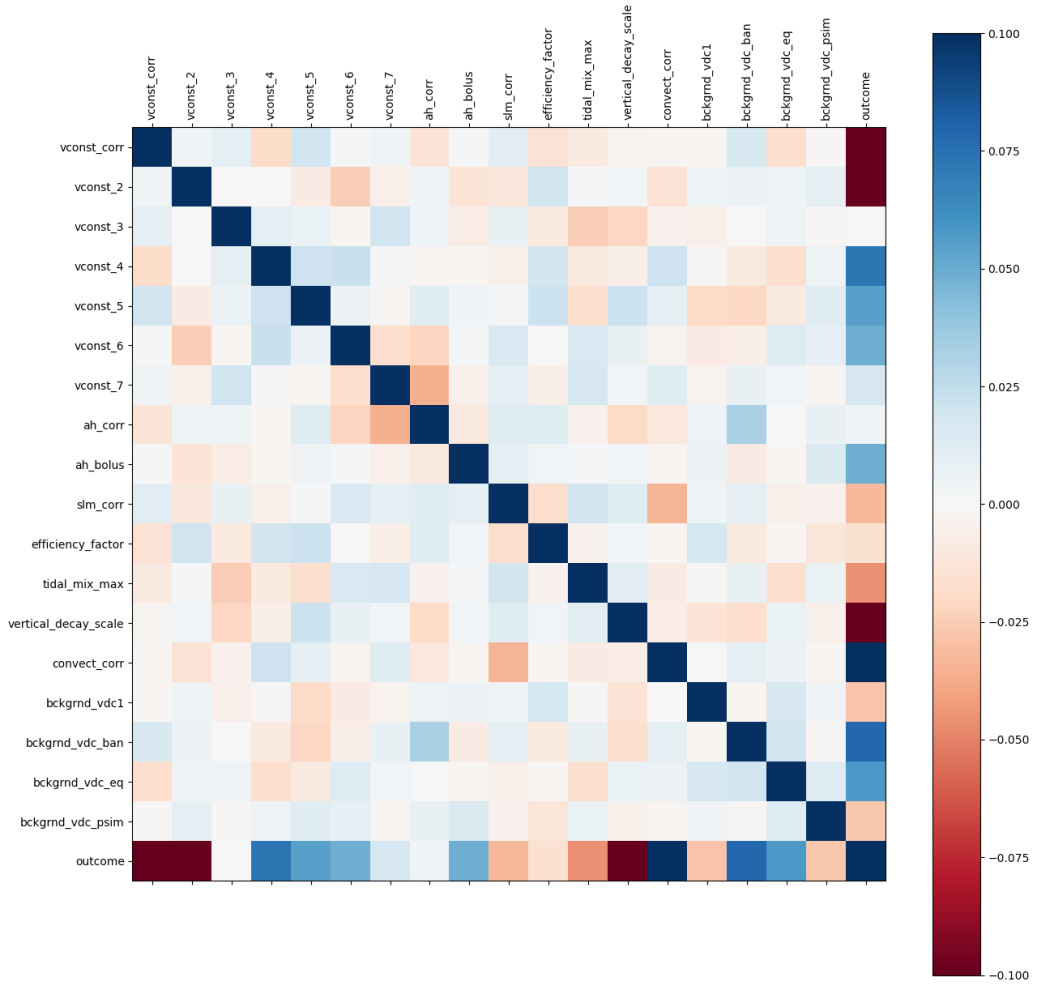
1 Przegląd danych



Rysunek 1: Rozkład danych

Pierwszą rzeczą jaką zauważamy na wykresie 1 jest to że cechy są od siebie całkowicie niezależne. Uniemożliwi to redukcję wymiarowości

Analizując postawowe parametry statystczne danych możemy zauważyć że dane są bardzo niezbalansowane - średnia z kolumny `outcome` wynosi **0.1**. co oznacza że 90% danych to `0` a jedynie 10% to `1`



Rysunek 2: Macierz korelacji

Na szczęscie macierz korelacji pokazuje że cechy są rzeczywiście zależne od `outcome` czego nie widać na wykresie 1 ponieważ `outcome` posiada jedynie dwa unikalne wartości.

2 Walidacja Modeli

W notatniku została przygotowana funkcja która jest użyta do walidowania modeli. Zostały tam wybrane niektóre metryki przedewszystkim:

- Accuracy
- Precision
- Recall
- R2
- ROC AUC

Metryką naważniejszą okazały sie `R2` i `ROC AUC` ponieważ `Accuracy` jest bardzo łatwo osiągalna przez modele i nie nadaje się do oceny ich jakości. Dodatkwo, przez dość małą ilość próbek metryki te są bardzo niestabilne Różnica nawet jednej próbki może zmienić wynik o kilkanaście procent w przypadku metryki R2 więc dodatkowo zostały załączone Metryki True Positive Rate i False Positive Rate i macierz pomyłek zarówno dla zbioru uczącego, testowego jak i zbalansowanego.

3 Użyte algorytmy i techniki

Aby poradzić sobie z tym problemem zostały przetestowane następujące algorytmy i techniki:

- Podstaowa regresja logistyczna
- Trenowanie na zbalansowanym podzbiorze
- Trenowanie na ważonym zbiorze
- Maszyny wektorów nośnych
- Polynomial Features
- Drzewa decyzyjne
- Regularyzowane sieci neuronowe

Do dobierania parametrów zostało użyte narzędzie `GridSearchCV` z biblioteki `scikit-learn`. Pozwala ono na przetestowanie wszystkich kombinacji parametrów i wybranie najlepszej z nich. Pozwoliło to szybko odfiltrować klasyfikatory które nie nadawały się do tego problemu - drzewa decyzyjne i generowanie cech wielomianowych. Dodatkowo pozwoliło to znaleść optymalne parametry dla SVM. Podczas eksperymentowania z sieciami neuronowymi również został użyte przeszukiwanie hiperparametrów ale bez skutku. Klasyfikatory które osiągneły słabe wyniki nawet w porównaniu do regresji logistycznej zostały zignorowane.

3.1 Przeszukiwanie hiperparametrów

Aby znaleść dobre klasyfikatory została zdefiniowana przestrzeń parametrów dla klasyfikatorów

- Regresja logistyczna
- Extra Trees Classifier
- Random Forest Classifier
- SVC

Wyniki przeszukiwania pokazały że jedynie model SVC z kernelm `linear` lub `rbf` osiągnął dobre wyniki. drzewa decyzyjne znalazły się wśród najlepszych ale nie osiągnęły wyników na poziomie SVM dlatego nie były dalej rozpatrywane

Została również wykonana cross-walidacja aby sprawdzić czy wyniki są stabilne. Modele SVM były niestabline, ale modele sieci neuronowych dawały raczej powtarzalne wyniki.

3.2 Sieci neuronowe

Klasyfikatory neuronowe zostały najpierw wstępnie przetestowane z różnymi architekturami:

- Wielowarstwowy klasyfikator z `Softmax`
  na końcu i funkcją straty `sparse_categorical_crossentropy`

- Wielowarstwowy klasyfikator z `Sigmoid` na końcu i funkcja straty `binary_crossentropy`
- Dropout na warstwach ukrytych
- Regularyzacja L1 i L2 na warstwie wejściowej
- Regularyzacja wyjścia inna inicjalizacja biasu
- Ważenie próbek wejściowych

Ostatecznie architektura zatrzymała się na trzech warstwach po 32 neurony, dropout 0.5, early stopping i funkcja aktywacji ELU.

Do uczenia zostały użyte sztuczki z tej strony link

4 Wyniki Klasyfikacji

Ogólnie bardzo dobrze poradziły sobie maszyny wektorów nośnych i sieci neuronowe, prawdopodobnie obie techniki osiągnęły maksymalny możliwy wynik R2 i AUC. Regresja logistyczna i drzewa decyzyjne osiągnęły wyniki owiele gorsze.

Szczegółowe wyniki można znaleść w notatniku `projekt.ipynb`. Tutaj zostaną przedstawione jedynie wybrane wyniki.

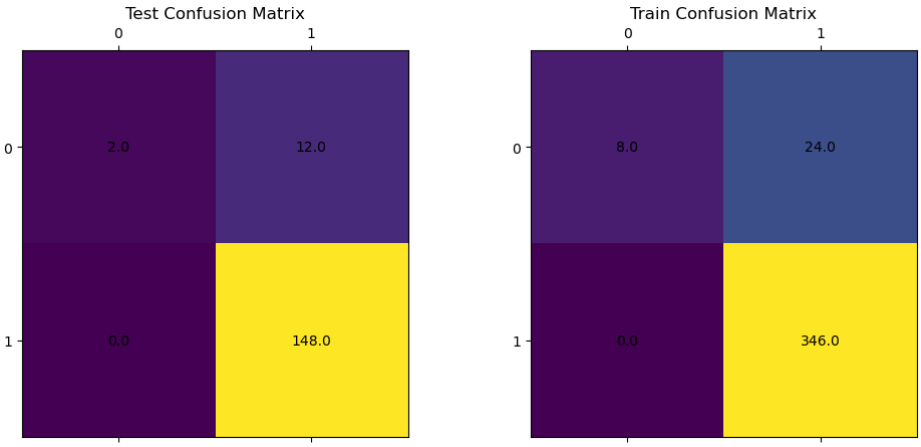
Na tabeli poniżej widac wszystkie modele które uzyskały w jakiś sposób satysfakcjonujący wynik, zazwyczaj oznaczało to wysoki wpsółczynnik R^2 i AUC, ale nie zawsze. W zależności od wymagań, można na przykład wybrać model generujacy większą dokładność danych albo model który ma większą pełność.

- `recall` - pełność
- `pre` - prezycja
- `acc` - dokładność
- `r2` - współczynnik determinacji

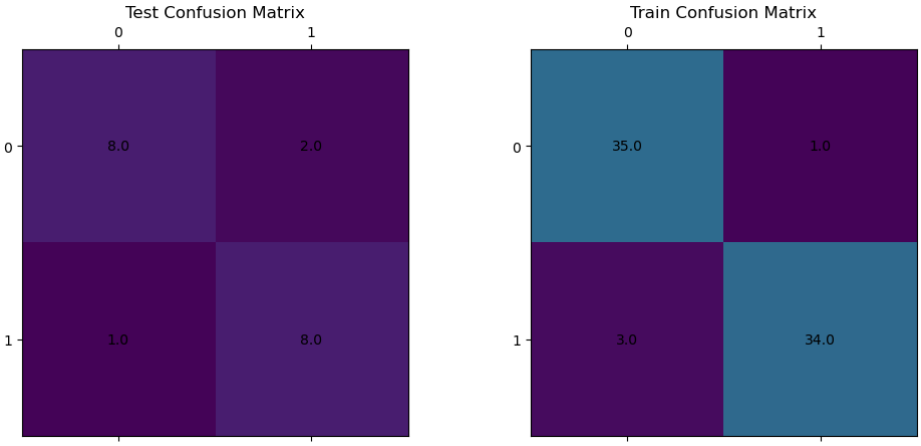
	recall		pre		acc		r2	
	val	train	val	train	val	train	val	train
Logistic Reg.	1	1	0.92	0.93	0.92	0.93	0.06	0.18
Logistic train bal.	0.88	0.91	0.80	0.93	0.84	0.92	0.36	0.78
SVM	0.99	0.99	0.97	0.97	0.96	0.97	0.60	0.65
SVM on balanced	1	1	0.75	0.79	0.84	0.86	0.36	0.50
SVM weighted	0.90	0.91	1	0.99	0.91	0.91	-0.09	-0.09
SVM w. on bal.	0.77	0.86	0.87	0.95	0.84	0.91	0.36	0.72
NN	0.97	0.98	0.97	0.97	0.96	0.96	0.53	0.55
NN on balanced	0.88	0.97	0.72	0.80	0.78	0.86	0.15	0.56
NN biased	0.97	0.98	0.99	0.98	0.96	0.97	0.60	0.65
NN b. on balaced	0.88	0.97	0.72	0.83	0.78	0.89	0.15	0.67
NN weighted	0.97	0.99	0.99	0.98	0.97	0.97	0.68	0.69
NN w. on balaced	0.88	0.97	0.72	0.83	0.78	0.89	0.15	0.67
Only ones	1	1	0.91	0.91	0.91	0.91	-0.09	-0.09
Perfect	1	1	1	1	1	1	1	1

Metryką na którą najbardziej zwracałem uwagę był R^2, ponieważ jest ona odporna na niezbalansowanie danych, ROC i AUC okazał się również dobrą metryką do przedwczesnego zatrzymywania uczenia. Ze względu na trudność problemu trzeba pujść na kompromis między dokładnością a pełnością.

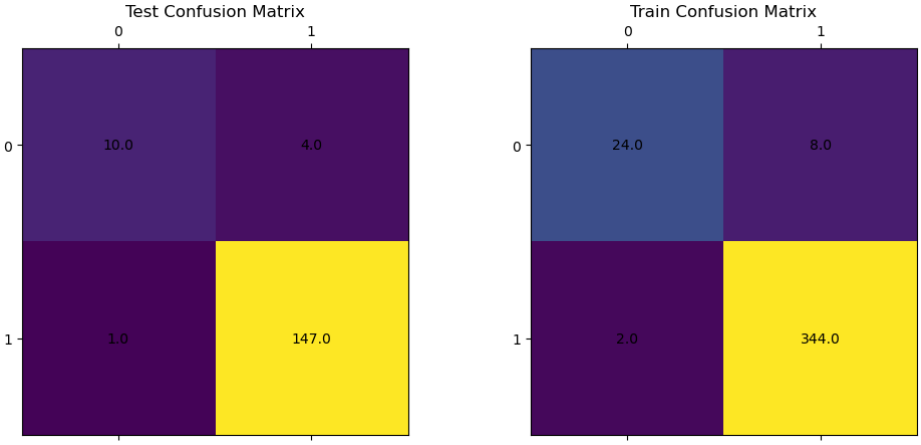
4.1 Macierze pomyłek



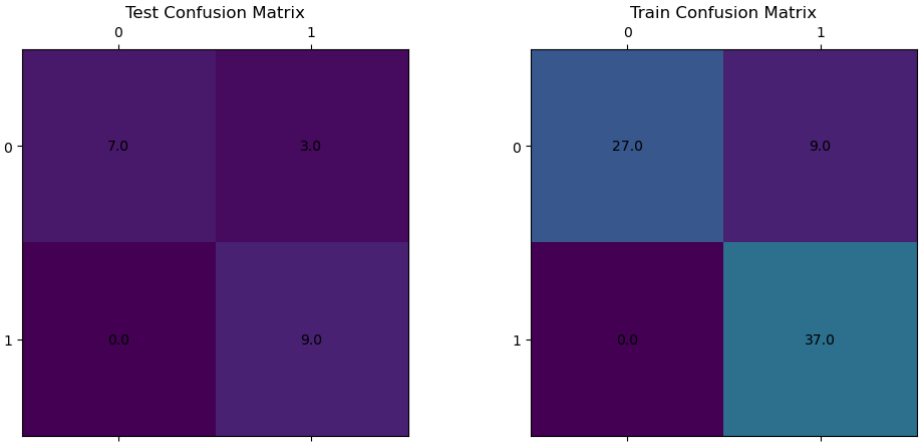
Rysunek 3: Logistic Reg.



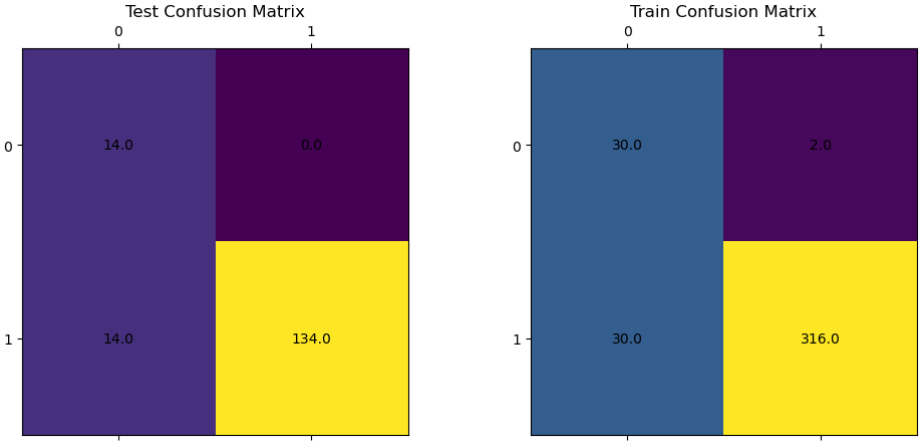
Rysunek 4: Logistic Reg. on balanced



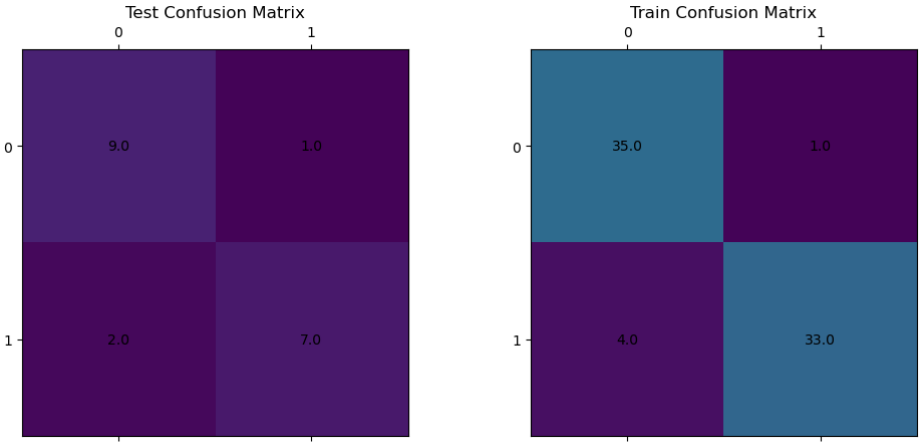
Rysunek 5: SVM



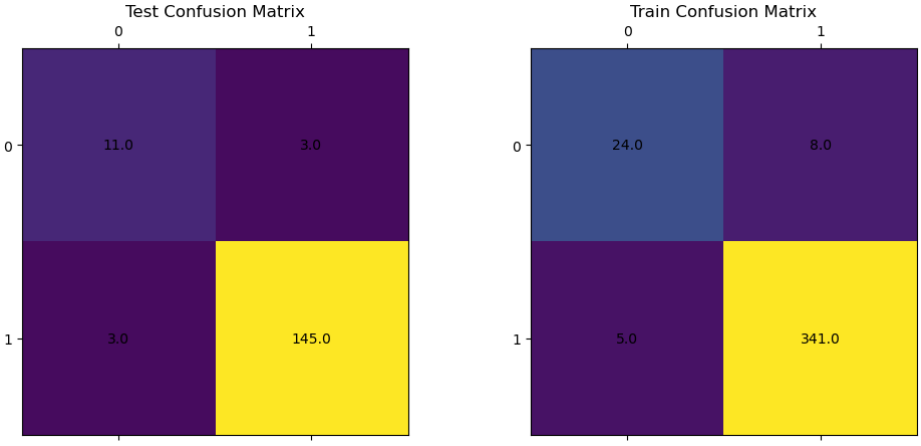
Rysunek 6: SVM on balanced



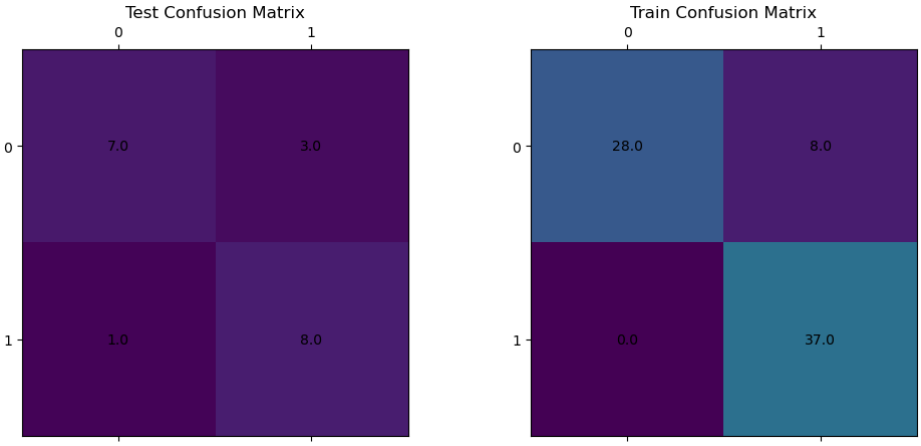
Rysunek 7: SVM weighted



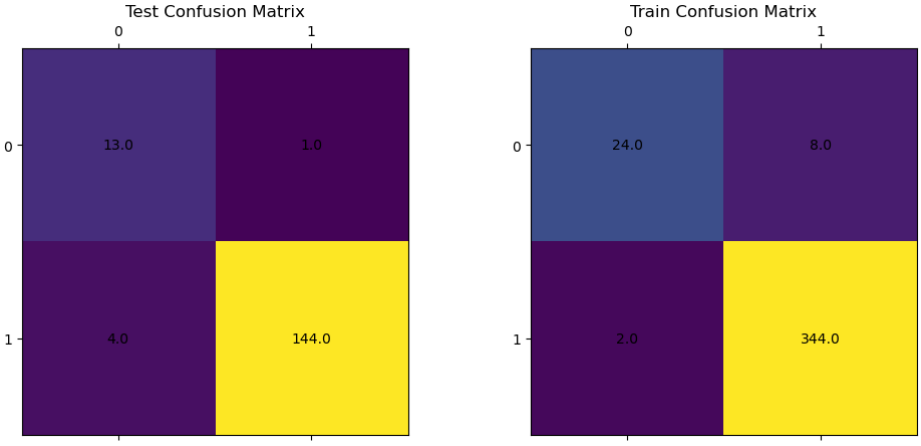
Rysunek 8: SVM weighted on balanced



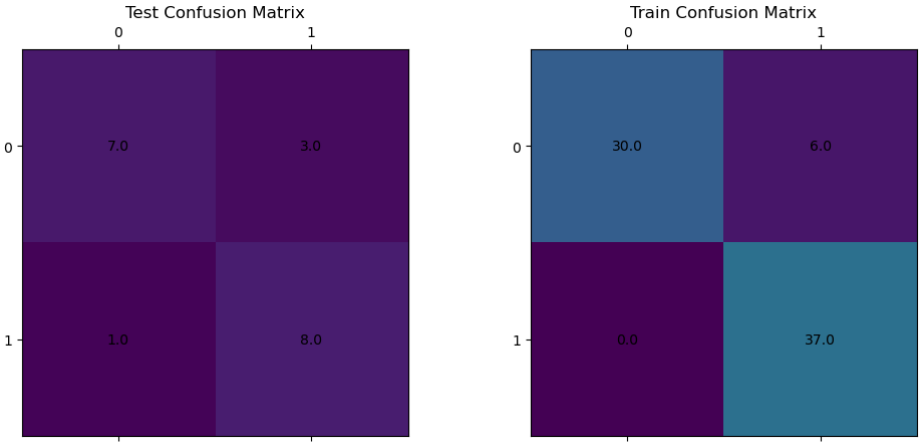
Rysunek 9: NN



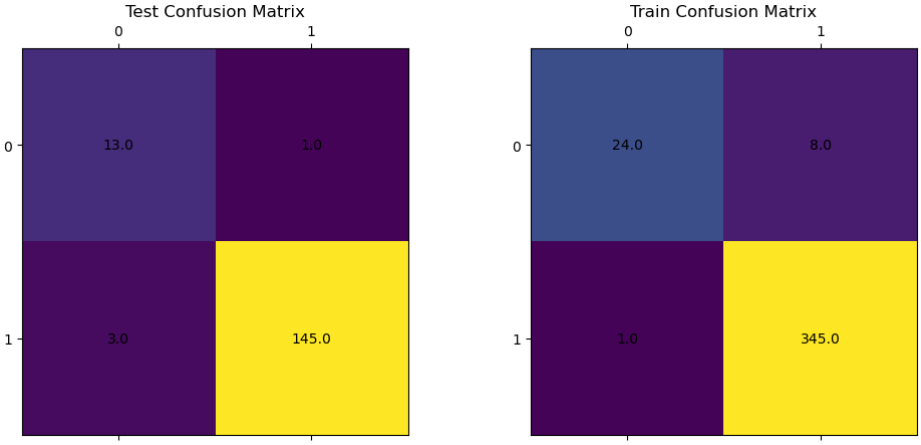
Rysunek 10: NN on balanced



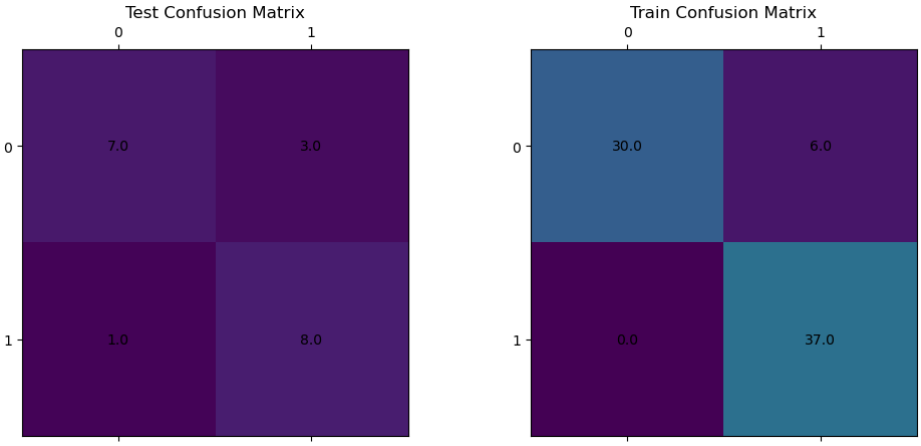
Rysunek 11: NN biased



Rysunek 12: NN biased on balanced



Rysunek 13: NN weighted



Rysunek 14: NN weighted on balanced

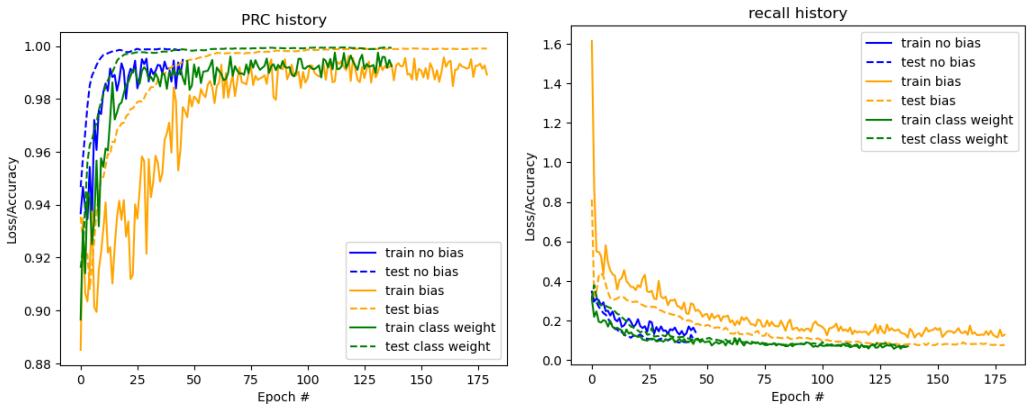
4.2 Macierze pomyłek opis

Należy zauważyć że modele bardzo często różnią się predukcją jednej/dwóch próbek. Na podstawie tych danych można znaleść kilka dobrych modeli

`SVM` radzi sobie dobrze uzyskując całkiem zadawalające wyniki, ale jest bardzo wrażliwy na zmiany w danych. Wersja z dostosowanymi wagami próbek również radzi sobie najlepiej przy danych 50/50.

`NN` radzi sobie w tym problemie bardzo dobrze i jest wstanie bez przetrenowania (utzymuje się na train i val) uzyskać bardzo wysoki wskaźnik r^2. Najlepszym modelem ze wszystkich jest właśnie sieć która posiada wagi dostosowane do ilości próbek, uzykuje r^2 na poziomie prawie 0.70 i to na wszystkich zbiorach! W zależności od oczekiwanych wyników można wybrać `Weighted SVM` ponieważ ilość fałszywie pozytywnych i fałszywie negatywnych odpowiedzi są do siebie zbliżone dla zbiorów testowych i zbalansowanych.

5 Krzywe uczenia



(a) prc (b) recall

Rysunek 15: Krzywe uczenia

Krzywe uczenia są dość obiecujące, ze względu na ostrą regularyzacje modelu przy uczeniu, błąd dla zbioru treningowego jest wyższy od zbioru testowego. A Błąd dla zbioru testowego bardzo szybko osiąga zbieżność co oznacza że model jest zmuszony szukać zależności w danych.

6 Podsumowanie

Uczenie modeli na taich danych jest okrutne i wymaga dużo strojenia hiperparametrów. Chociaż z odpowiednimi technikami jest się w stanie uzyskać dobre wyniki. W tym przypadku najlepszym modelem jest sieć neuronowa z dostosowanymi wagami pozwlającymi uczyć się na niezbalansowanych zbiorach danych lub SVM.

Dużym wyzwaniem jest przetestowanie modeli ponieważ nie ma możliwości uzyskania dokładnych wyników

Dokładne wyniki dostępne są w notatniku.

Literatura

[1] set-up-the-end-to-end-trainingevaluation-skeleton–get-dumb-baselines

[2] Uczenie maszynowe z użyciem Scikit-Learn i Tensorflow Aurelien Geron Wydanie inicjalizacja

[3] scikit-learn model selection