Przegląd modeli klasyfikacji i metod preprocessingu w problemie identyfikacji rodzaju szkła

Karolina Łukasik

18 czerwca 2023

Spis treści

1 Wstęp

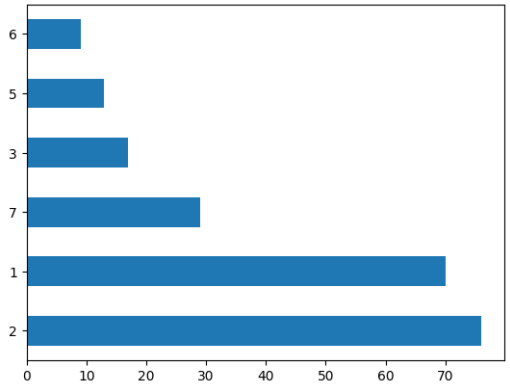
W poniższym raporcie przedstawimy przegląd wybranych modeli klasyfikacji dla problemu identyfikacji szkła. Dodatkowo przetestujemy różne metody preproccesingu danych i ich wpływ na wyniki osiągane przez modele. Postaramy się również dobrać hiperparametry modeli w taki sposób, by zmaksymalizować osiągane wyniki.

2 Analiza danych

Zbiór danych składa się z 214 rekordów oraz 9 cech. Osiem z tych cech opisuje procentowy udział danego pierwiastka w próbce szkła. Dodatkowo mamy informacje na temat współczynnika załamania światła RI (Refractive Index). W zbiorze danych nie ma brakujących wartości. Każdy z rekordów należy do jednego z 7 typów szkła, ponumerowanych liczbą od 1 do 7. W praktyce nie ma żadnego rekordu należacego do typu 4. By lepiej zwizualizować sobie dane przyjrzyjmy się kilku rekordom, budowie ramki danych, a następnie liczności danych typów w zbiorze.

	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	TypeOfGlass
0	1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0.00	0.0	1
1	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0.00	0.0	1
2	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0.00	0.0	1
3	1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0.00	0.0	1
4	1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0.00	0.0	1

Rysunek 1: Ramka danych



Rysunek 2: Liczność typów

	1	2	3	4	5	6	7
liczność	70	76	17	0	13	9	29

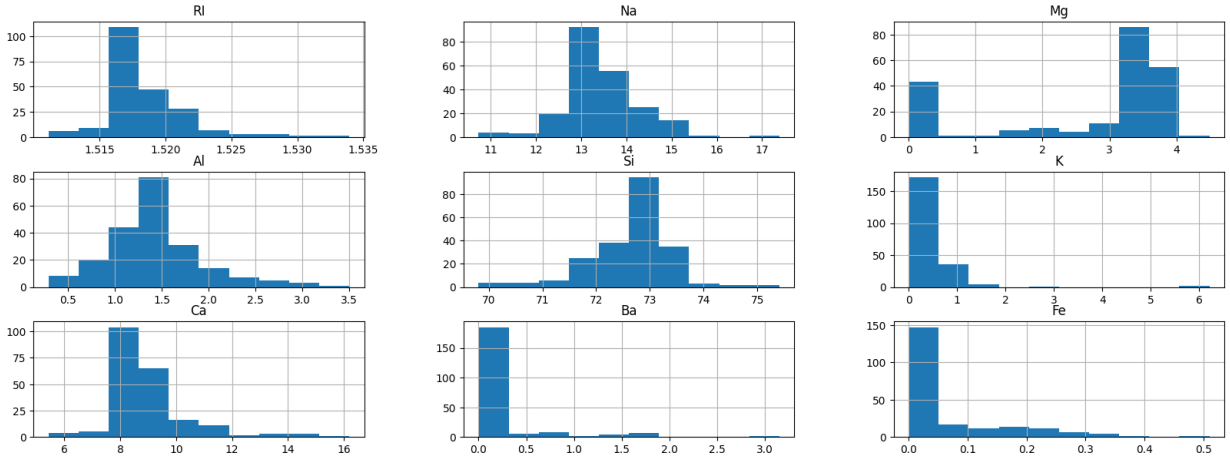
Tabela 1: Liczności typów

Na podstawie powyższego wykresu oraz tabeli widać, że dane są bardzo niezbalansowane ze znaczną przewagą przedstawicieli typu szkła 1 i 2. Musimy więc zadbać o to by w zbiorach treningowych znalazły się rekordy z każdej kategorii. Użyjemy w tym celu *StratifiedKFold*.

Idąc dalej możemy się przyjrzeć rozkładom cech. Pomocne w tym celu mogą się okazać histogramy jak i podstawowe statystyki.

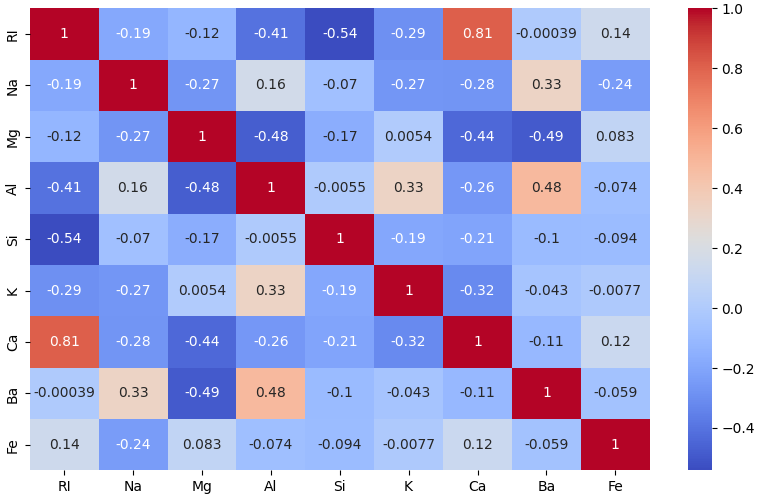
	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe
count	214.000000	214.000000	214.000000	214.000000	214.000000	214.000000	214.000000	214.000000	214.000000
mean	1.518365	13.407850	2.684533	1.444907	72.650935	0.497056	8.956963	0.175047	0.057009
std	0.003037	0.816604	1.442408	0.499270	0.774546	0.652192	1.423153	0.497219	0.097439
min	1.511150	10.730000	0.000000	0.290000	69.810000	0.000000	5.430000	0.000000	0.000000
25%	1.516522	12.907500	2.115000	1.190000	72.280000	0.122500	8.240000	0.000000	0.000000
50%	1.517680	13.300000	3.480000	1.360000	72.790000	0.555000	8.600000	0.000000	0.000000
75%	1.519157	13.825000	3.600000	1.630000	73.087500	0.610000	9.172500	0.000000	0.100000
max	1.533930	17.380000	4.490000	3.500000	75.410000	6.210000	16.190000	3.150000	0.510000

Rysunek 3: Podstawowe statystyki



Rysunek 4: Histogramy poszczególnych cech

Z powyższych rysunków możemy wywnioskować, iż szkło w każdym przypadku składa się głównie z siarki, dalej z sodu i wapnia. Poza tym pierwiasti Fe, K oraz Ba w dużej większości w ogóle nie pojawiają się w szkle. Dodatkowo takie cechy jak RI oraz zawartość Fe cechują się bardzo małą wariancją.



Rysunek 5: Korelacje pomiędzy cechami

Na koniec analiz możemy się przyjrzeć diagramowi korelacji, na którym widać, jak poszczególne cechy są powiązane ze sobą. Głównie w oczy rzuca się wysoka korelacja pomiędzy Ca, a współczynnikiem RI. Może być to wartościowa informacja przy decyzji o eliminacji cech. Może się bowiem okazać, że wystarczyłoby w zbiorze pozostawić tylko jedną z tych cech.

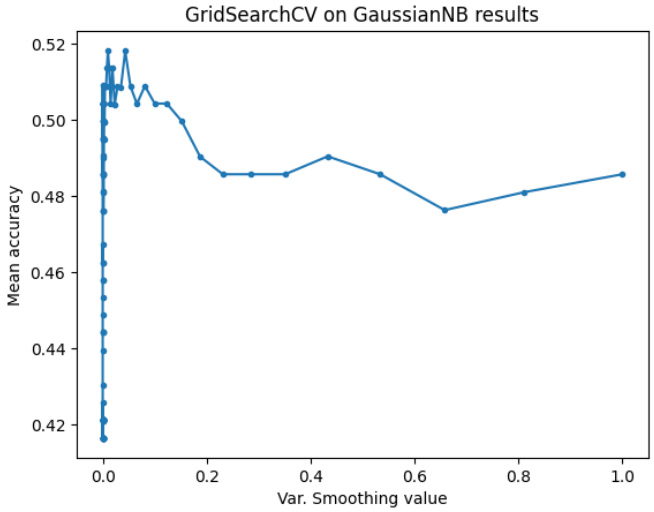
3 Tuning hiperparametrów

Metody, których użyjemy do zbudowania modeli to:

- Naiwny Klasyfikator Bayesa
- Drzewo Decyzyjne
- Las Losowy
- Regresja Logistyczna
- Support Vector Classifier

Za pomocą metody GridSearchCV możemy znaleźć najlepsze wartości hiperparametrów tzn. takie, dla których modele osiągną najlepsze wyniki. Przyjrzyjmy się wynikom dla każdego z modeli:

3.0.1 Naiwny Klasyfikator Bayesa



Rysunek 6: Tuning hiperparametru dla Naiwnego Bayesa

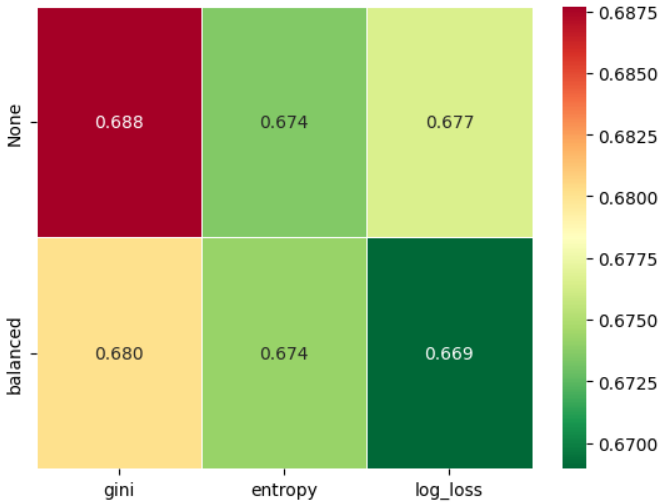
Najlepszy wynik osiągnięty został dla parametru *var_smoothing=0.01*.

3.0.2 SVC

Najlepszym zestawem parametrów dla Support Vector Classifiera okazało się:

- C = 1000
- gamma = 0.01
- kernel = rbf

3.0.3 Drzewo decyzyjne



Rysunek 7: Tuning hiperparametrów dla Drzewa Decyzyjnego

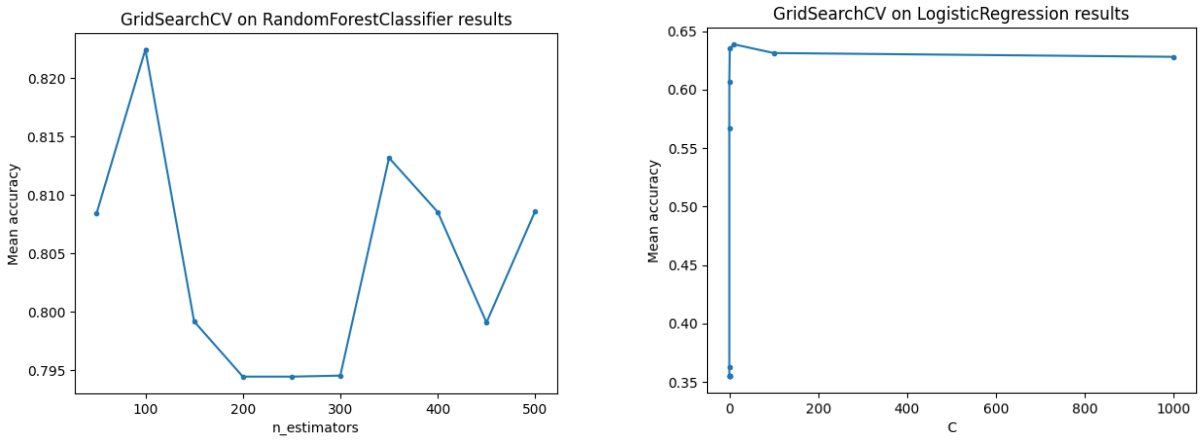
W przypadku drzewa decyzyjnego najlepsze okazało się pozostawienie domyślnego kryterium *gini* oraz niebalansowanie wag.

3.0.4 Random Forest

Dla modelu Random Forest modyfikacja liczby drzew nie zmieniała znacznie osiąganych wyników. Najlepszy wynik osiągnęła jednak dla $n\_estimators$=100.

3.0.5 Regresja logistyczna

Najlepszą wartością parametru okazało się C=10.



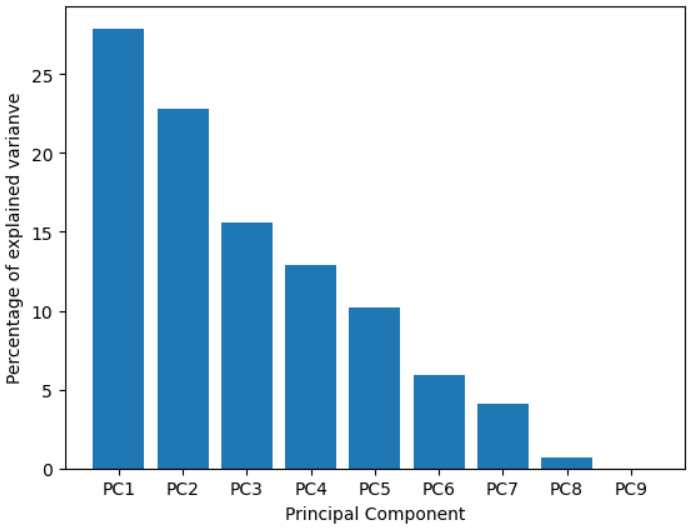
Rysunek 8: Tuning hiperparametrów dla Random Forest oraz Regresji Logistycznej

4 Preprocessing

4.1 PCA - Principal Component Analysis

W celu zmniejszenia wymiaru zbioru danych często stosuje się technikę PCA. Polega ona na liniowej transformacji danych do nowego układu współrzędnych i stworzeniu nowych cech, które w malejący procentowo sposób reprezentują swój wkład w wariacje danych.

Zastosowałam tę technikę na naszym zbiorze i otrzymałam następujące cechy:



Rysunek 9: Komponenty PCA

Widać, że każde kolejne cechy mają duże znaczenie w opisie danych. Aby zachować około 95% informacji o zbiorze, musimy zostawić przynajmniej 6 pierwszych komponentów.

4.2 Brakujące wartości

Mimo, iż w zbiorze nie ma obecnych brakujących wartości, możemy sztucznie usunąć np. 5% wartości z każdej z kolumn i znaleźć strategię na wypełnianie tych wartości.

W celu usunięcia wartości wylosowałam dla każdej z kolumn 10 wartości z przedziału od 0 do 213 i zmieniłam wartości w odpowiednim wierszu dla danej kolumny na *NaN*. Powstała ramka danych (samych cech) prezentowała się więc mniej więcej tak:

	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe
5	1.51596	12.79	3.61	1.62	72.97	0.64	8.07	0.0	0.26
6	1.51743	13.30	3.60	NaN	73.09	0.58	8.17	0.0	0.00
7	1.51756	13.15	3.61	1.05	73.24	0.57	8.24	NaN	0.00
8	1.51918	14.04	3.58	1.37	72.08	0.56	8.30	0.0	0.00
9	NaN	13.00	3.60	1.36	72.99	0.57	8.40	0.0	NaN
10	1.51571	12.72	3.46	1.56	73.20	0.67	8.09	0.0	0.24
11	1.51763	12.80	3.66	1.27	NaN	0.60	8.56	0.0	0.00
12	1.51589	12.88	3.43	1.40	73.28	0.69	8.05	0.0	0.24
13	1.51748	12.86	3.56	1.27	73.21	0.54	8.38	0.0	0.17

Rysunek 10: Zbiór danych z brakującymi wartościami

Metodą którą użyłam później do wypełnienia brakujących wartości był *KNNImputer*, czyli wyznaczenie wartości cechy na podstawie średniej tego parametru dla kilku najbliższych sąsiadów tego rekordu.

4.3 Skalowanie

W następnej sekcji przedstawię porównanie działania modeli z wykorzystaniem trzech różnych sposobów skalowania danych numerycznych w opozycji do nieskalowania danych w ogóle. Techniki, które zastosowałam, To:

- Standaryzacja
- Normalizacja
- skalowanie MinMax, czyli liniowe przekształcenie na zbiór [0,1]

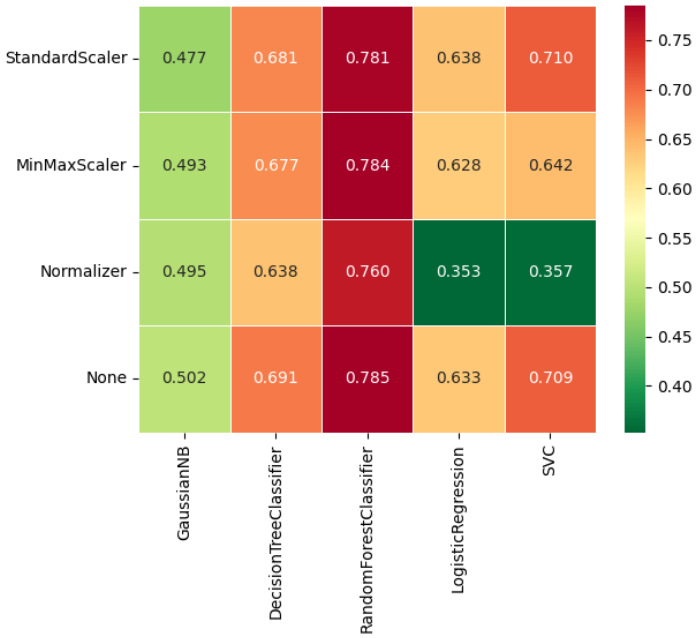
5 Trenowanie modeli

5.1 Pipeline

W celu złożenia skalowania, wypełniania brakujących wartości z PCA oraz wybranym modelem klasyfikacji, stworzyłam funkcję budującą Pipeline z wybranych metod. Tworzy ona osobne transformery dla kolumn numerycznych i kategorycznych z odpowiednimi metodami preprocessingu. Dalej opcjonalnie dodaje redukcję wymiaru PCA. Możemy wytrenować taki pipeline na kolejnych klasyfikatorach i sposobach skalowania.

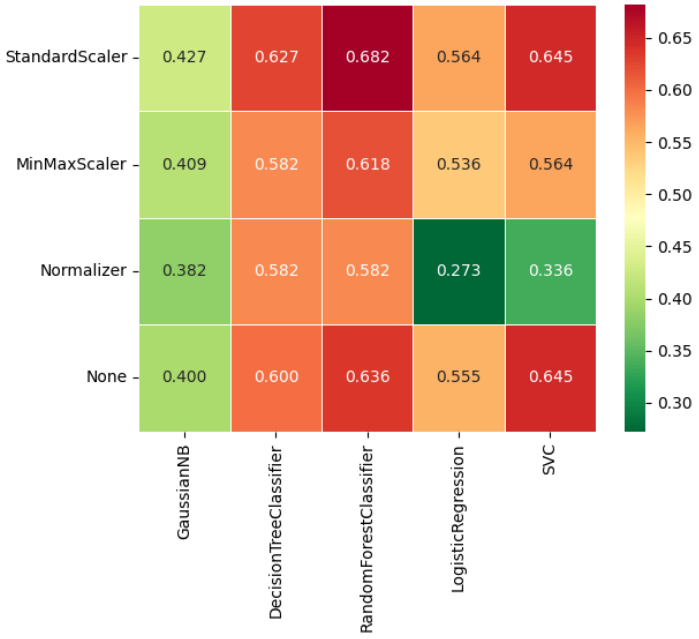
5.2 Porównanie modeli

Na poniższym rysunku przedstawiona jest heatmapa dokładności, jaką osiągnęły modele na różnych metodach skalowania. Same modele wytrenowane zostały na hiperparametrach, które osiągnęły najlepsze wyniki w sekcji 3. Metodą, którą użyłam do cross validacji było *RepeatedStratifiedKFold*.



Rysunek 11: Porównanie modeli

Możemy również porównać modele przy zastosowaniu redukcji wymiaru PCA i pozostawieniu 6 komponentów.



Rysunek 12: Porównanie modeli po PCA

W tym wypadku otrzymaliśmy jednak gorsze wyniki.

6 Metryki oceny modeli

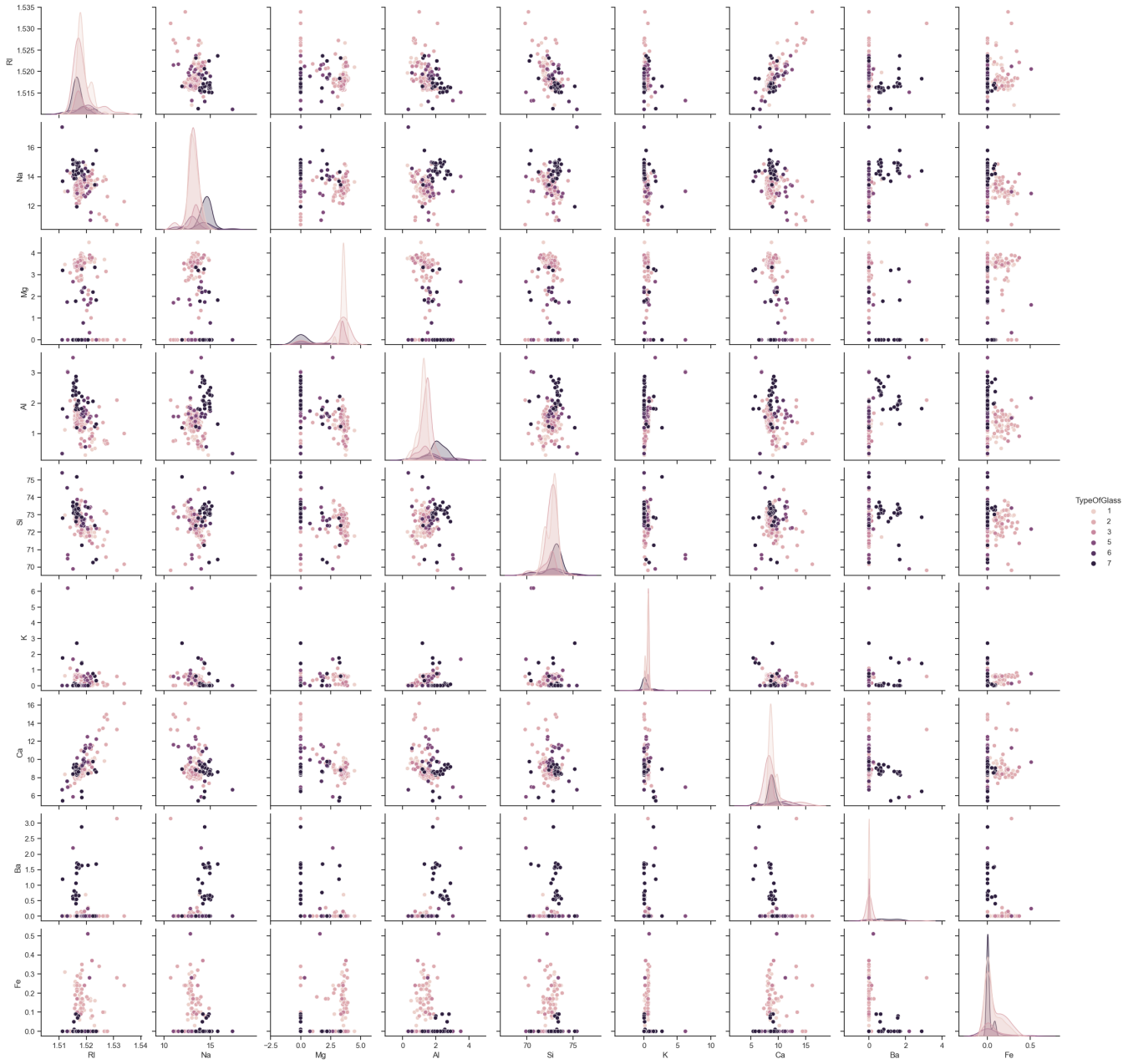
Mając wybrane najlepsze hiperparametry oraz metody skalowania dla każdego z modeli, możemy przejść do bardziej szczegółowego ich porównania. Wcześniej porównywaliśmy jedynie dokładność, teraz przyjrzymy się również metrykom: precision, recall oraz f1_score. Utrudnieniem jest tutaj fakt, iż nie mamy tylko dwóch klas, więc musimy wziąć średnią wartość wymienionych metryk z każdej z klas. W tym celu wytrenowałam każdy z modeli za pomocą najlepszych dla nich parametrów na tym samym podziale na zbiór treningowy i testowy. Osiągnęły one następujące wyniki:

	accuracy	mean precision	mean recall	mean f1 score
Random Forest	0.77	0.737	0.682	0.693
Naive Bayes	0.47	0.435	0.477	0.44
Decision Tree	0.67	0.705	0.635	0.655
Logistic Regression	0.56	0.485	0.517	0.5
SVC	0.7	0.698	0.647	0.655

Tabela 2: Liczności typów

Wyraźnie widać, iż najlepszym modelem do predykcji typu szkła okazał się Random Forest. Tuż za nim z nieco gorszymi wynikami uplasował się SVC oraz Drzewo Decyzyjne. Z drugiej strony Regresja Logistyczna oraz Naive Bayes osiągnęły zdecydowanie gorsze, niezadawalające wyniki.

7 Dodatek: zależności między parami cech



Rysunek 13: Wyskresy rozrzutu

8 Użyte biblioteki

Przy realizacji zadania korzystałam głównie z biblioteki sklearn oraz pomocniczo pandas, numpy, matplotlib i seaborn. Z biblioteki sklearn skrzystałam z metod cross validacji, modeli, metryk oceny, metod preprocessingu, PCA oraz Pipeline'a.