

# Raport projektowy – Ocena jakości wina czerwonego Vinho Verde

## 1. Przedstawienie problemu.

Celem projektu było przygotowanie modelu uczenia maszynowego, który będzie w stanie z zadawalającą skutecznością klasyfikować wina czerwone gatunku Vinho Verde z północy Portugalii do kategorii jakościowych (od 1 do 10) na podstawie ich właściwości chemicznych.

## 2. Dane.

W zestawie danych wejściowych znalazło się 1599 obserwacji posiadających 11 zmiennych wejściowych i jedną zmienną wyjściową.

Lista zmiennych wejściowych:

1. Kwasowość stała (g/L)
2. Kwasowość lotna (g/L)
3. Zawartość kwasu cytrynowego (g/L)
4. Zawartość cukru resztowego (g/L)
5. Zawartość chlorków (g/L)
6. Zawartość wolnego dwutlenku siarki (mg/L)
7. Zawartość całkowitego dwutlenku siarki (mg/L)
8. Gęstość (g/cm<sup>3</sup>)
9. pH (wartości od 0 do 14)
10. Zawartość siarczanów (g/L)
11. Zawartość alkoholu (%)
12. Jakość (ocena od 0 do 10).

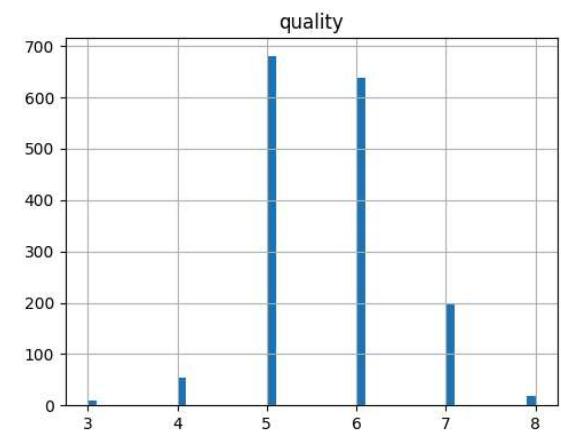
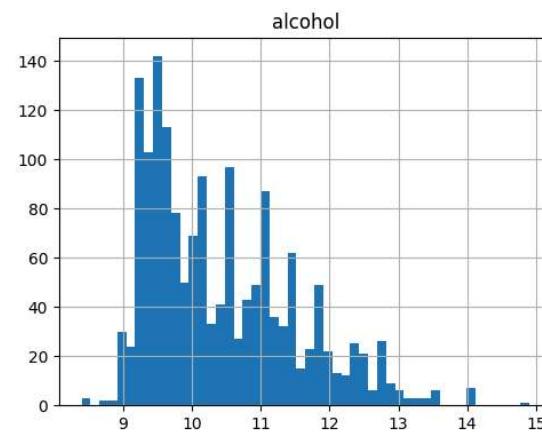
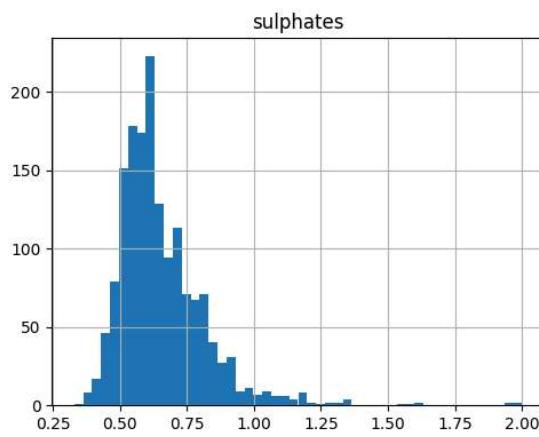
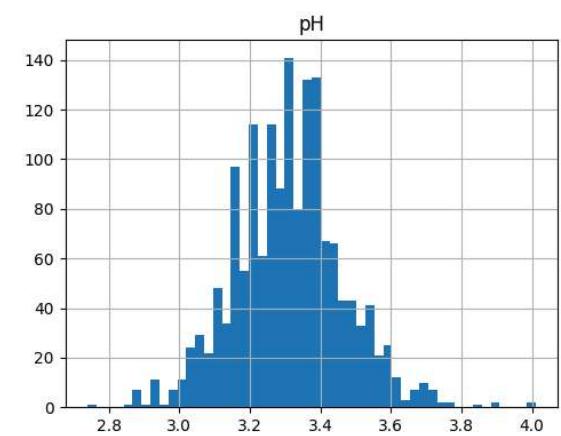
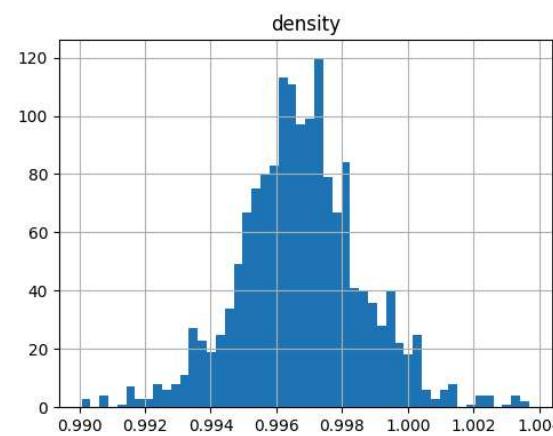
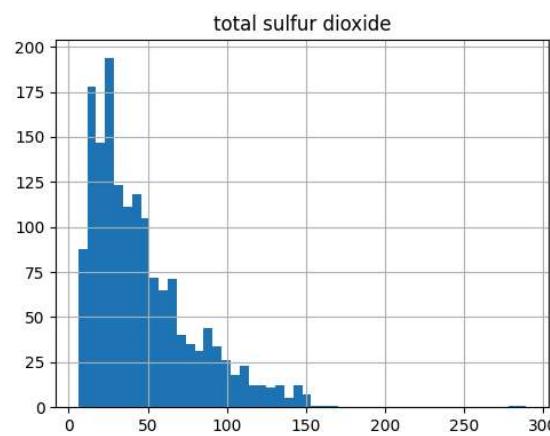
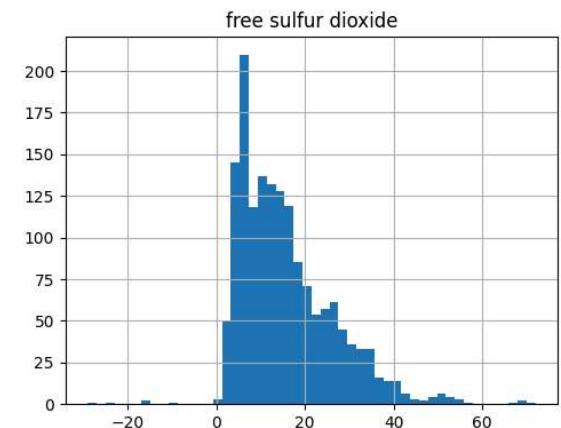
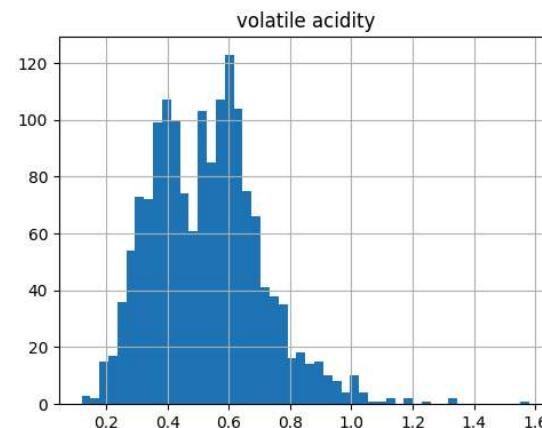
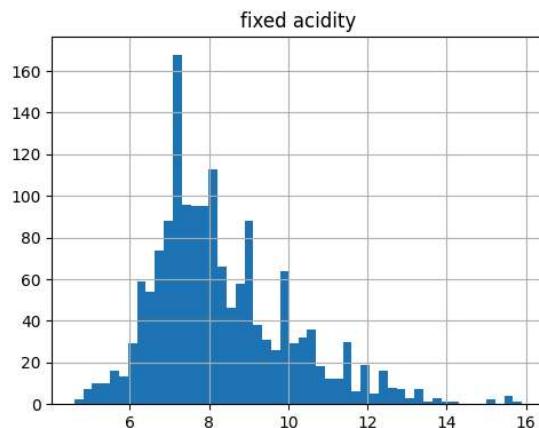
Zmienna wyjściowa:

1. Jakość (wartości od 1 do 10)

Link do danych: [Dane](#)

Rozkład zmiennych został przedstawiony na poniższych histogramach:

```
In [ ]: import rapport_functions  
data = rapport_functions.get_starting_data()  
rapport_functions.display_histogram(data)
```



Podstawowym problemem napotkanym w danych była duża rozbieżność pod względem ilość rekordów należących do poszczególnych klas jakościowych. Klasy 1, 2, 9 i 10 nie posiadały żadnych rekordów, a klasy 3, 4 i 8 posiadały ich niewiele. Zbiór danych był zdecydowanie zdominowany przez klasy 5, 6 i 7.

Podstawowe informacje o zbiorze danych:

```
In [ ]: rapport_functions.display_info_data(data)
```

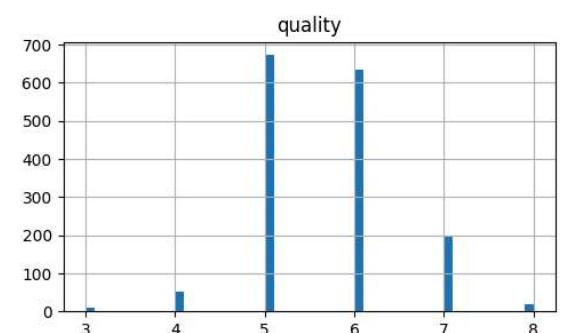
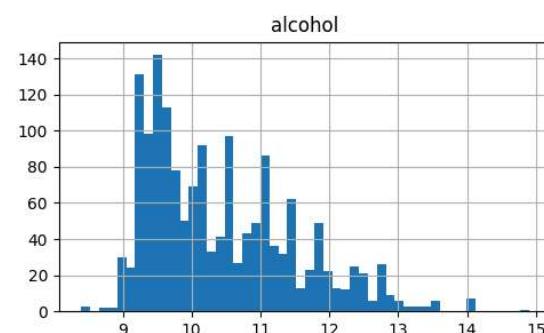
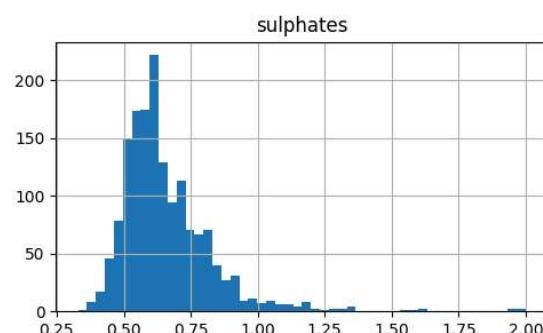
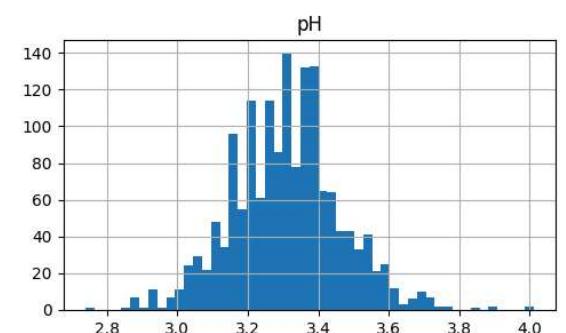
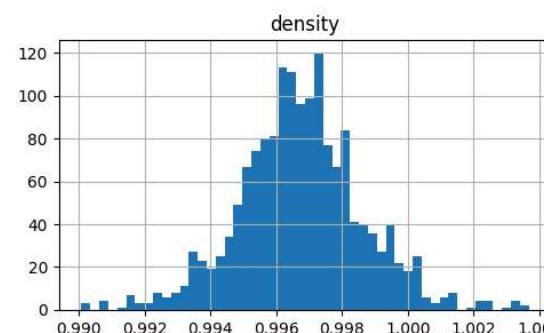
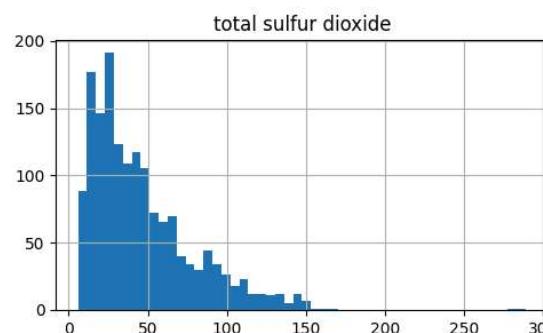
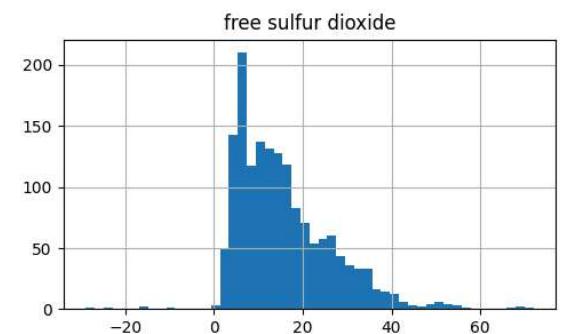
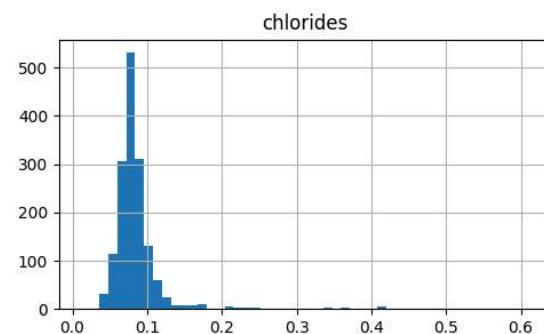
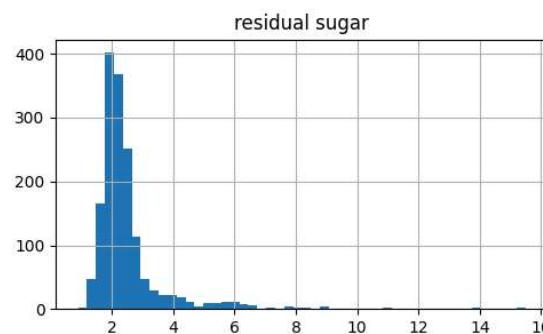
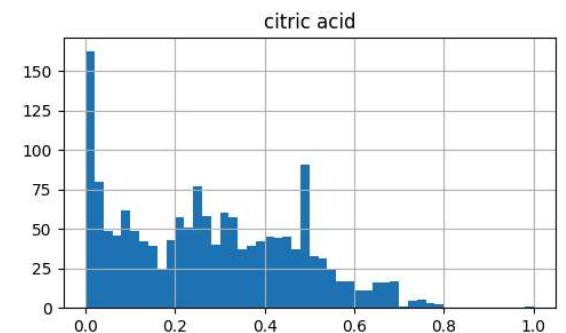
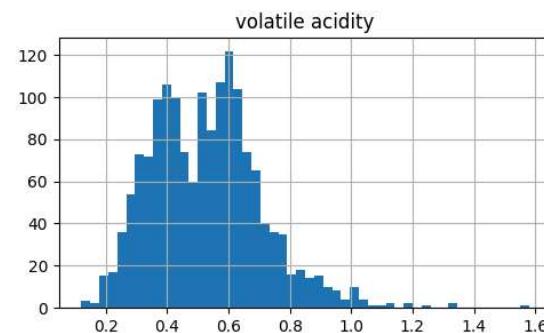
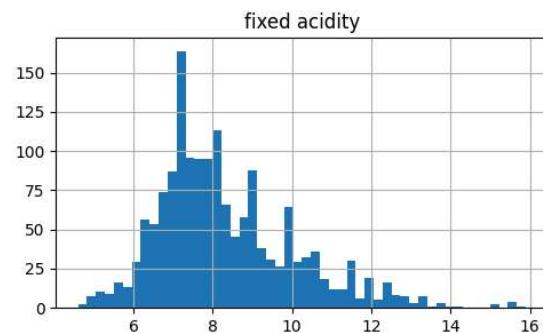
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   fixed acidity    1599 non-null   float64 
 1   volatile acidity 1599 non-null   float64 
 2   citric acid     1599 non-null   object  
 3   residual sugar   1599 non-null   object  
 4   chlorides        1599 non-null   object  
 5   free sulfur dioxide 1599 non-null   float64 
 6   total sulfur dioxide 1599 non-null   float64 
 7   density          1595 non-null   float64 
 8   pH               1599 non-null   float64 
 9   sulphates        1599 non-null   float64 
 10  alcohol          1599 non-null   float64 
 11  quality          1599 non-null   int64  
dtypes: float64(8), int64(1), object(3)
memory usage: 150.0+ KB
None
```

Następnie dane zostały poddane procesowi pre-processingu. Wykonano następujące kroki:

1. Pozbyto się zbędnych znaków z wartością danych wejściowych.
2. Przekonwertowano typ danych dla wszystkich kolumn na numeryczny.
3. Odrzucono wszystkie rekordy posiadające puste wartości w którejkolwiek z kolumn.
4. Odrzucono wszystkie rekordy z wartościami skrajnymi w którejkolwiek z kolumn.

W wyniku czego w zbiorze danych pozostały 1503 rekordy.

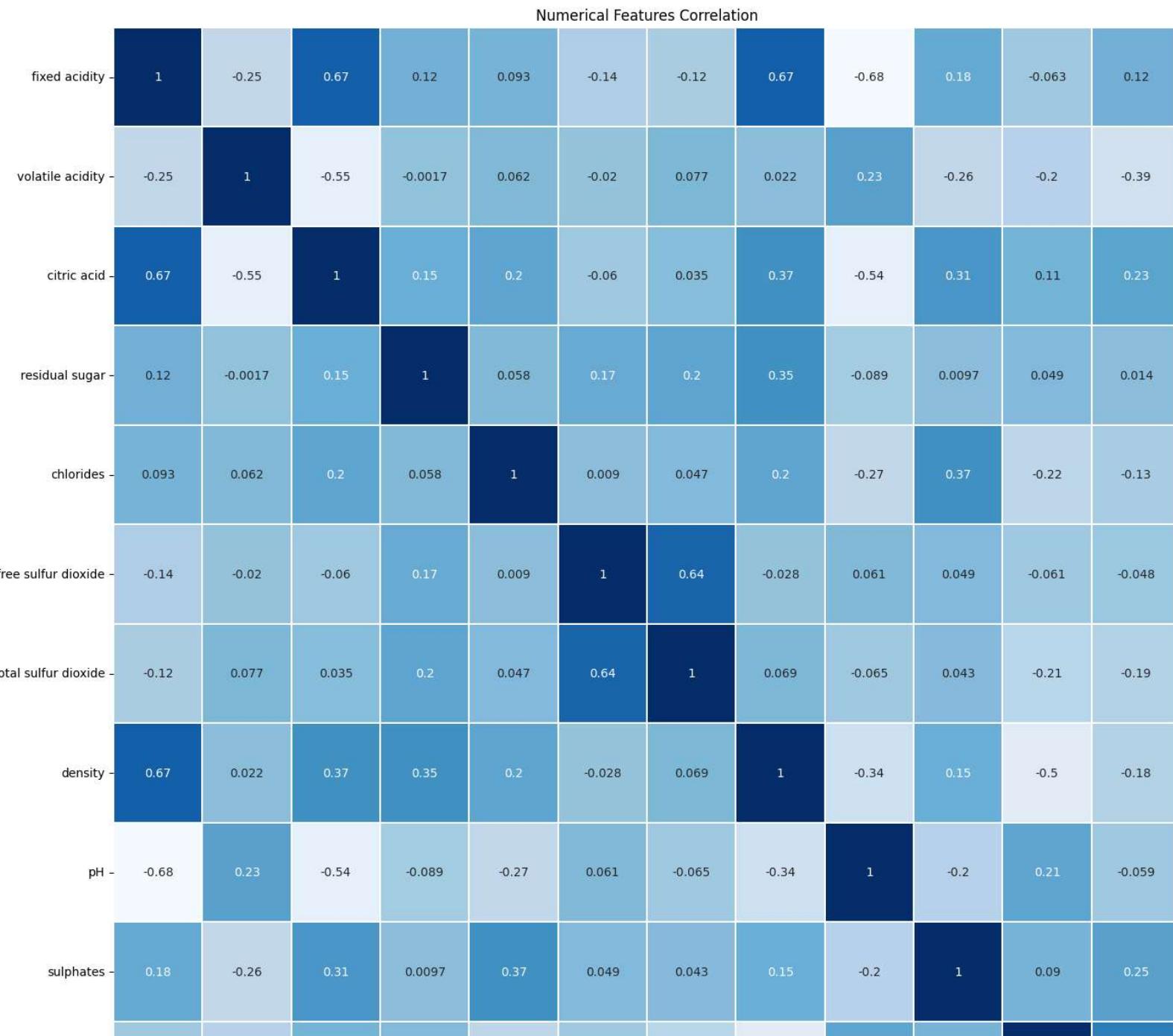
```
In [ ]: rapport_functions.modify_data_initial(data)
rapport_functions.display_histogram(data)
rapport_functions.display_info_data(data)
```

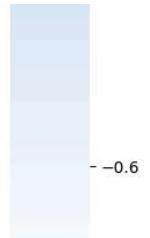


```
<class 'pandas.core.frame.DataFrame'>
Index: 1588 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype  
---  --  
 0   fixed acidity      1588 non-null   float64 
 1   volatile acidity   1588 non-null   float64 
 2   citric acid        1588 non-null   float64 
 3   residual sugar     1588 non-null   float64 
 4   chlorides          1588 non-null   float64 
 5   free sulfur dioxide 1588 non-null   float64 
 6   total sulfur dioxide 1588 non-null   float64 
 7   density            1588 non-null   float64 
 8   pH                 1588 non-null   float64 
 9   sulphates          1588 non-null   float64 
 10  alcohol            1588 non-null   float64 
 11  quality            1588 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 161.3 KB
None
```

Odrzucono także cechy o niskiej relevancji statystycznej. Za pułap, dla którego cecha została uznawana za istotną statystycznie, przyjęto korelację ze zmienną wynikową o wartości większej 0.1.

```
In [ ]: rapport_functions.corr_matrix(data)
```





alcohol -	-0.063	-0.2	0.11	0.049	-0.22	-0.061	-0.21	-0.5	0.21	0.09	1	0.47
quality -	0.12	-0.39	0.23	0.014	-0.13	-0.048	-0.19	-0.18	-0.059	0.25	0.47	1
fixed acidity -												
volatile acidity -												
citric acid -												
residual sugar -												
chlorides -												
free sulfur dioxide -												
total sulfur dioxide -												
density -												
pH -												
sulphates -												
alcohol -												
quality -												

Cechy które pozostały w zbiorze danych to:

1. Kwasowość stała (g/L).
2. Zawartość kwasu cytrynowego (g/L).
3. Zawartość chlorków (g/L).
4. Zawartość całkowita dwutlenku siarki (mg/L).
5. Gęstość (g/cm<sup>3</sup>).
6. Zawartość siarczanów (g/L).
7. Zawartość alkoholu (%).

## 2. Model 1 (benchmark) - Naive Bayes Classifier

Cechy modelu:

- Łatwy do implementacji i interpretacji
- Skalowalny
- Jest w stanie uchwycić złożone zależności
- Podatny na bycie kierowanym przez klasy dominujące

Kroki wykonane w ramach tworzenia modelu:

1. Inicjalizacja i przeskalowanie danych za pomocą StandardScalera.

2. Podział danych na zestaw treningowy i testowy w proporcji 4:1, co pozostawiło nas z 1202 rekordami z zbiorze treningowym i 301 w zbiorze testowym.
3. Wybranie strategii próbkowania.
4. Utworzenie obiektu klasy GaussianNB.
5. Uczenie i ewaluacja modelu o standardowych hiperparametrach przy wykorzystaniu walidacji krzyżowej (do celów poglądowych).
6. Definicja zbioru hiperparametów do zoptymalizowania.
7. Wyznaczenie optymalnych hiperparametrów.
8. Uczenie i ewaluacja modelu o standardowych hiperparametrach przy wykorzystaniu walidacji krzyżowej.
9. Wytrenowanie modelu o zoptymalizowanych parametrach.
10. Wykonanie przez model predykcji na zbiorze testowym.

```
In [ ]: #Wyswietlenie raportu klasifikacji, optymalnych hiperparametrów, macierzy pomyłek i innych przydatnych wykresów.  
raport_functions.gaussian_naive_bayes_main()
```

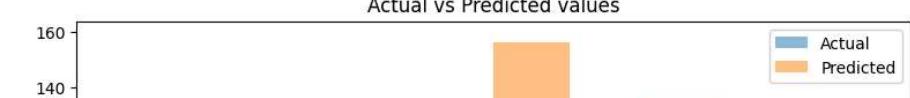
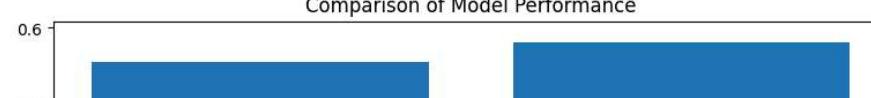
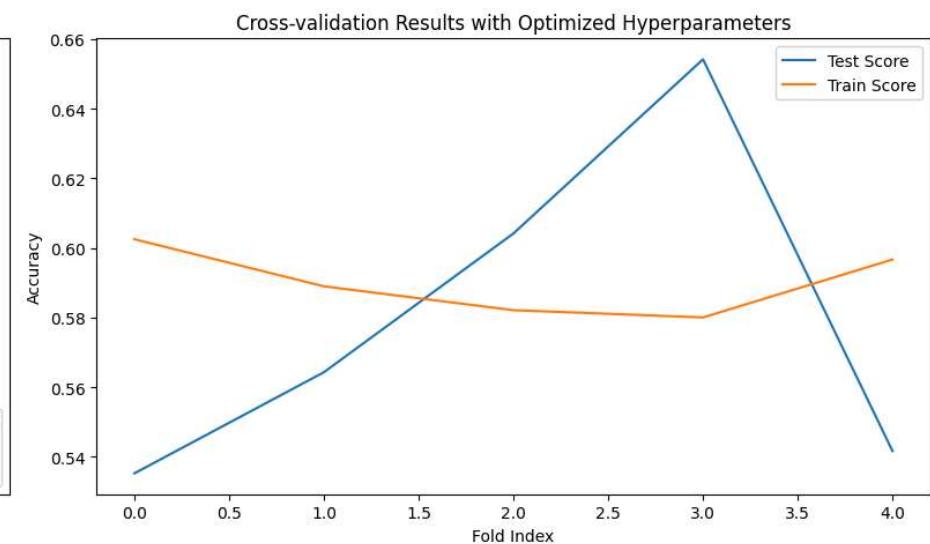
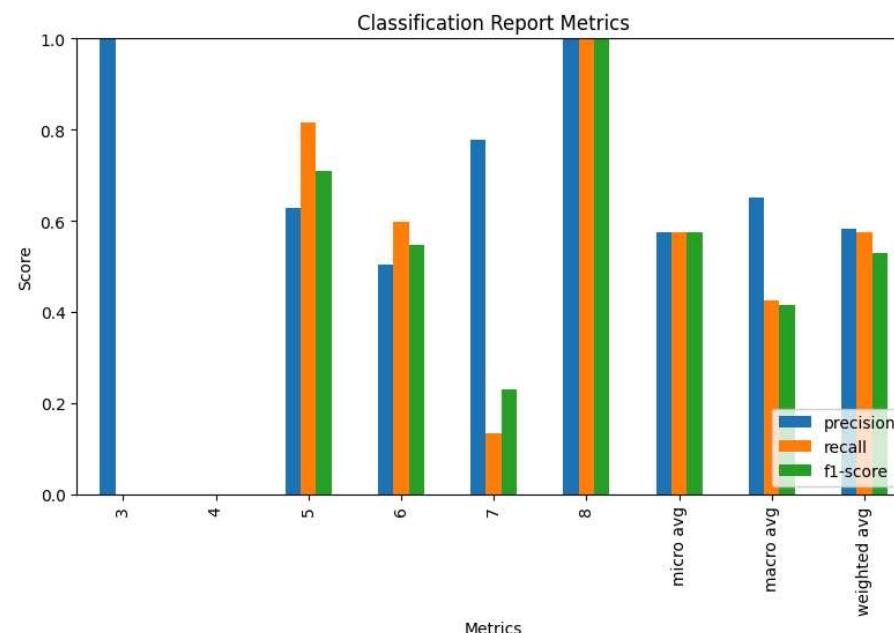
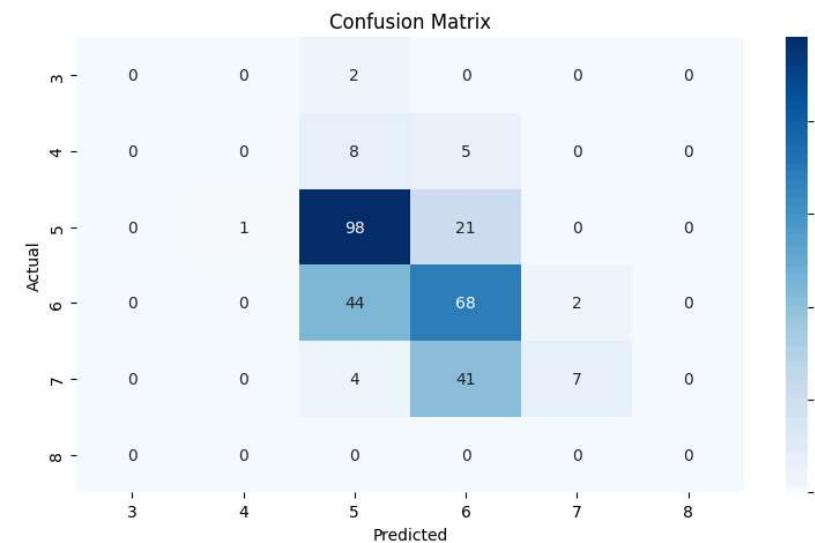
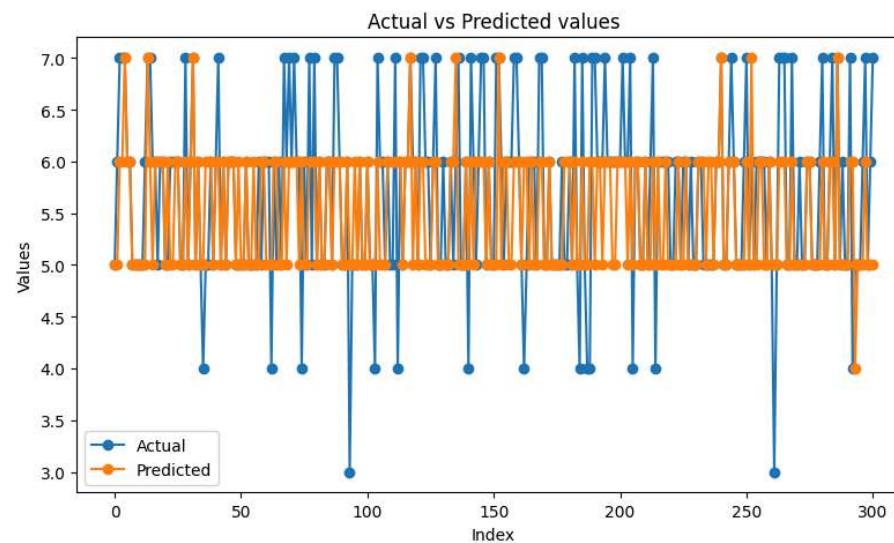
```
2024-06-06 00:11:17,223 - INFO - Best parameters: {'var_smoothing': 0.43287612810830584}
```

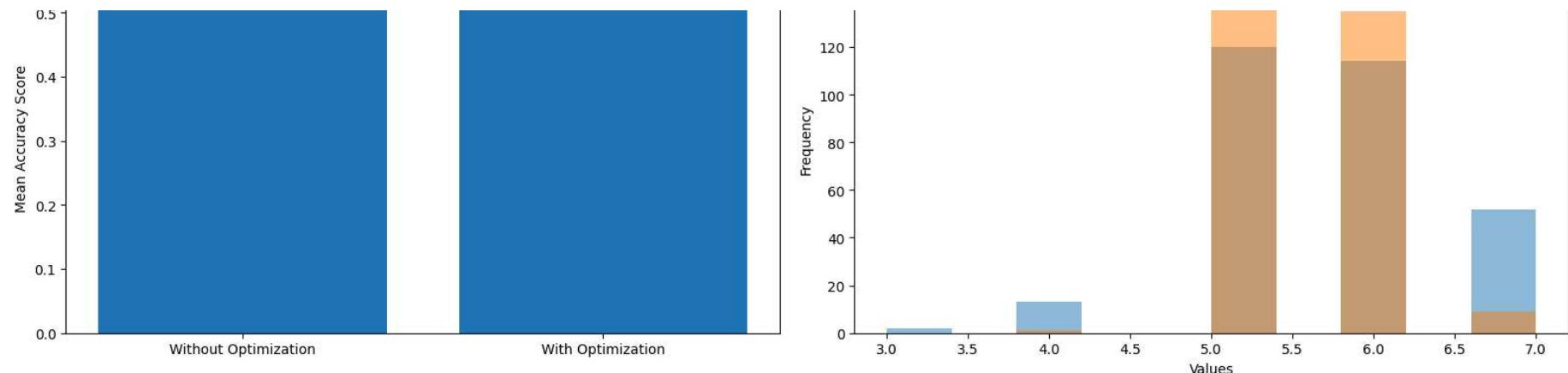
```
2024-06-06 00:11:17,250 - INFO - Classification report:
```

	precision	recall	f1-score	support
3	1.00	0.00	0.00	2
4	0.00	0.00	0.00	13
5	0.63	0.82	0.71	120
6	0.50	0.60	0.55	114
7	0.78	0.13	0.23	52
8	1.00	1.00	1.00	0
micro avg	0.57	0.57	0.57	301
macro avg	0.65	0.42	0.41	301
weighted avg	0.58	0.57	0.53	301

```
Fitting 5 folds for each of 50 candidates, totalling 250 fits
```

```
0.574750830564784
```





## Ocena modelu:

### Optymalne hiperparametry

Za optymalne hiperparametry spośród podanych do wyznaczenia uznano:

- var\_smoothing: 0.43287612810830584

### Wyniki modelu

- model osiągnął dokładność na poziomie 57%,
- klasą o najwyższym poziomie F1 była klasa 5, miara ta wyniosła dla niej 0.71,
- miał tendencję do zbyt częstego w stosunku do danych rzeczywistych klasyfikowania win jako klasę 5,
- miał tendencję do zbyt częstego w stosunku do danych rzeczywistych przydzielania win do klas 5 i 6, za to klasa 4 i 7 cierpały na niedobór zaklasyfikowanych win w stosunku do stanu rzeczywistego,
- wraz ze spadkiem liczby przedstawicieli danej klasy jakościowej w zbiorze danych, spadała także skuteczność modelu w klasyfikowaniu win do tychże klas.

### Podsumowanie

Model nie poradził sobie dobrze z problemem klasyfikacji. Zarówno miara dokładności, jak i czułości, pozostawiają wiele do życzenia.

### 3. Model 2 - KNN

Cechy modelu:

- Potrafi osiągać dużą dokładność w warunkach dobrze oznakowanych i wolnych od szumów danych, co ma miejsce w naszym przyodku.
- Algorytm K-NN jest wyjątkowo dobry w rozpoznawaniu skomplikowanych wzorców w danych, takich jak wielowymiarowe cechy chemiczne wina, które mogą być subtelnie powiązane z jakością wina na dyskretnej skali od 1 do 10.
- W przeciwieństwie do innych technik klasyfikacyjnych, K-NN nie wymaga założeń statystycznych dotyczących rozkładu danych. Jest to korzystne w przypadku rzeczywistych, często nieskonsolidowanych zestawów danych, co zwiększa jego uniwersalność i adaptacyjność do różnorodnych zbiorów danych.
- K-NN pozwala na łatwą zmianę metryki dystansu (np. euklidesowa, Manhattan), co umożliwia dostosowanie algorytmu do specyfiki danych wina, które mogą różnić się skalą i naturą. Ta elastyczność jest kluczowa, gdy dane różnią się w zakresie ich skali lub wymagają niestandardowego traktowania dystansu.

Kroki wykonane w ramach tworzenia modelu:

1. Inicjalizacja i przeskalowanie danych za pomocą MinMaxScalera.
2. Podział danych na zestaw treningowy i testowy w proporcji 4:1, co pozostawiło nas z 1202 rekordami w zbiorze treningowym i 301 w zbiorze testowym.
3. Wybranie strategii próbkowania.
4. Utworzenie obiektu klasy KNeighborsClassifier.
5. Uczenie i ewaluacja modelu o standardowych hiperparametrach przy wykorzystaniu walidacji krzyżowej (do celów poglądowych).
6. Definicja zbioru hiperparametów do zoptymalizowania.
7. Wyznaczenie optymalnych hiperparametrów.
8. Uczenie i ewaluacja modelu o standardowych hiperparametrach przy wykorzystaniu walidacji krzyżowej.
9. Wytrenowanie modelu o zoptymalizowanych parametrach.
10. Wykonanie przez model predykcji na zbiorze testowym.

```
In [ ]: #wyświetlenie raportu klasyfikacji, optymalnych hiperparametrów, macierzy pomyłek i innych przydatnych wykresów.  
raport_functions.knn_main()
```

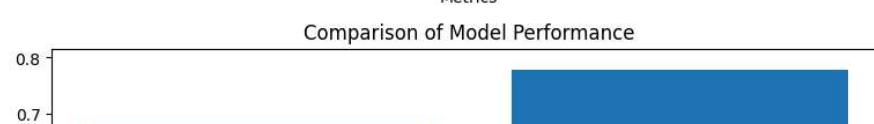
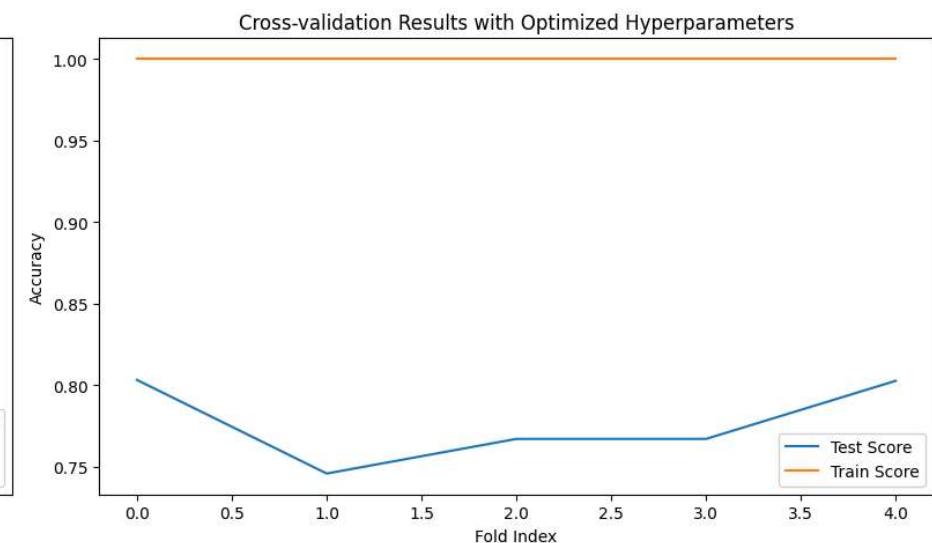
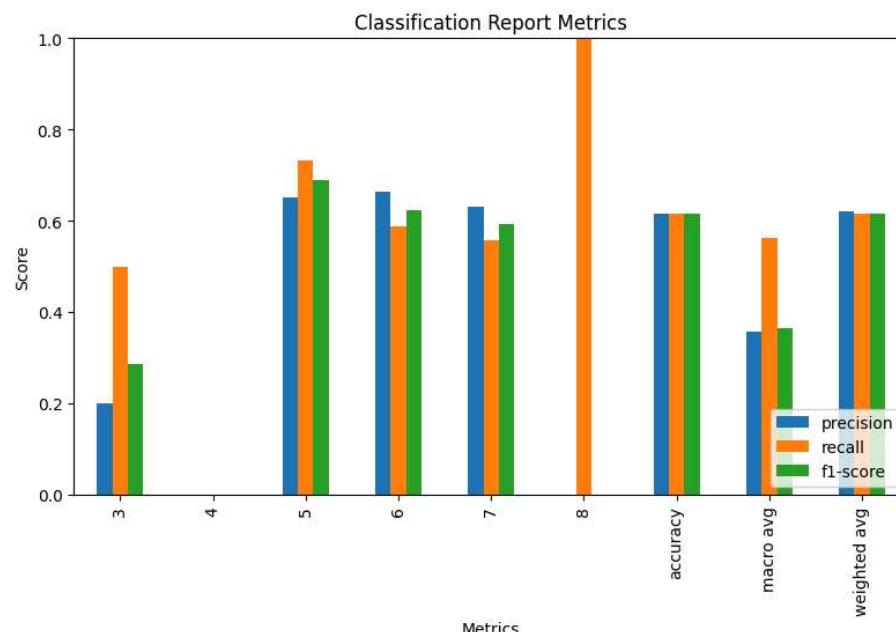
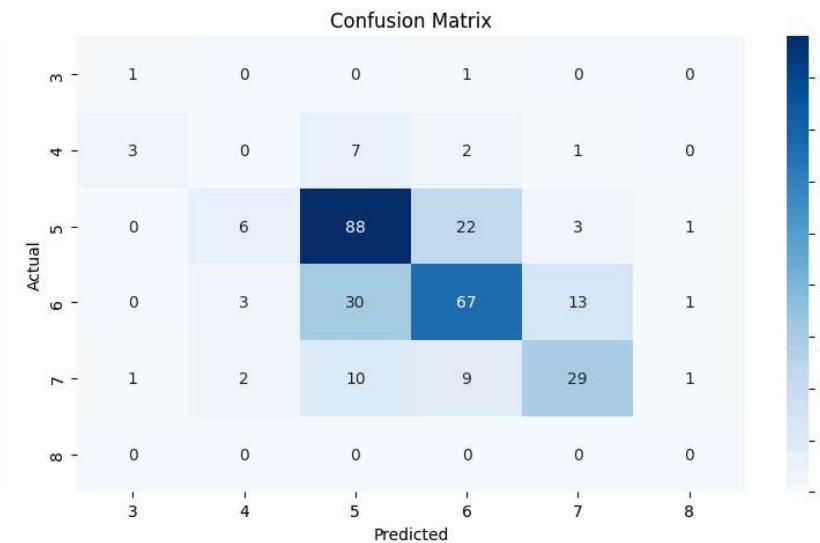
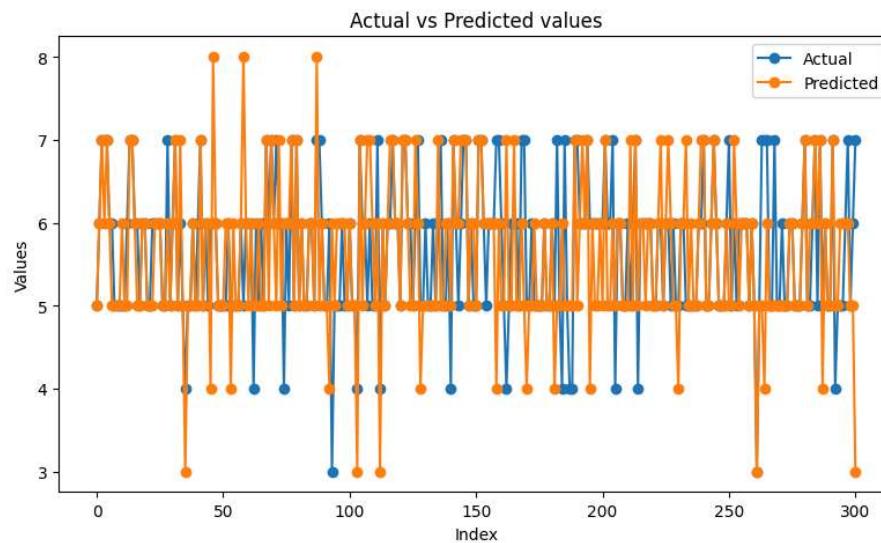
```
2024-06-06 00:11:18,453 - INFO - Data initialization and splitting complete.  
Fitting 5 folds for each of 50 candidates, totalling 250 fits
```

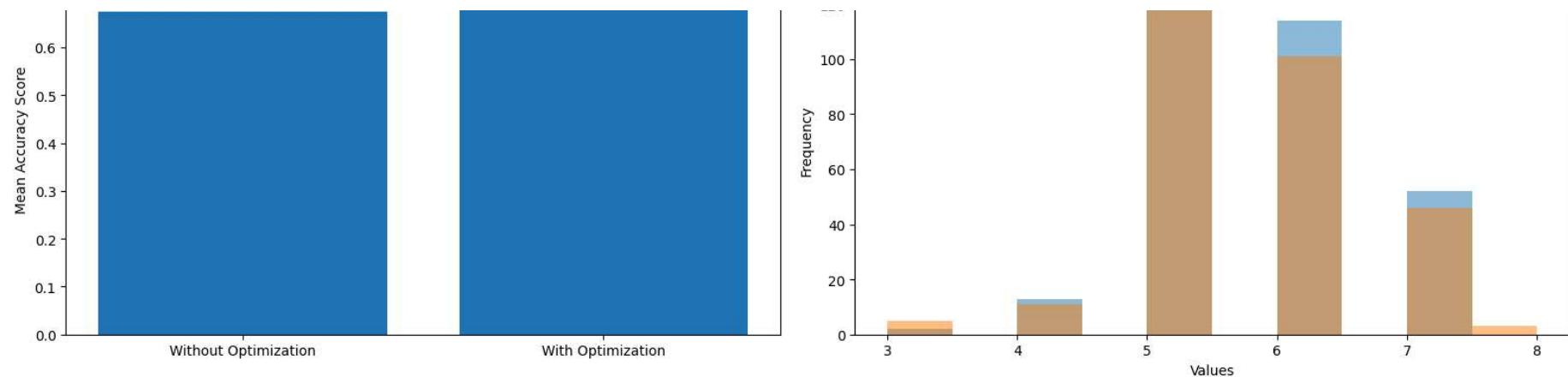
```
2024-06-06 00:11:19,839 - INFO - Best parameters: {'weights': 'uniform', 'p': 1, 'n_neighbors': 1, 'algorithm': 'brute'}
2024-06-06 00:11:19,945 - INFO - Classification report:
precision    recall    f1-score   support

      3       0.20      0.50      0.29       2
      4       0.00      0.00      0.00      13
      5       0.65      0.73      0.69     120
      6       0.66      0.59      0.62     114
      7       0.63      0.56      0.59      52
      8       0.00      1.00      0.00       0

  accuracy                           0.61      301
  macro avg       0.36      0.56      0.37      301
weighted avg       0.62      0.61      0.62      301
```

0.6146179401993356





## Ocena modelu:

### Optymalne hiperparametry

Za optymalne hiperparametry spośród podanych do wyznaczenia uznano:

- weights: uniform,
- p: 1,
- n\_neighbors: 1,
- algorithm: brute

## Wyniki modelu

- model osiągnął dokładność na poziomie 61%,
- klasą o najwyższym poziomie F1 była klasa 5, miara ta wyniosła dla niej 0.69. Oprócz niej, pułap 0.60 osiągnęła klasa 6,
- miał tendencję do zbyt częstego w stosunku do danych rzeczywistych przydzielania win do klasy 5, za to klasy 6 i 7 cierpiąła na niedobór zaklasyfikowanych win w stosunku do stanu rzeczywistego,
- wraz ze spadkiem liczby przedstawicieli danej klasy jakościowej w zbiorze danych, spadała także skuteczność modelu w klasyfikowaniu win do tychże klas,
- model wykazuje wyraźnie oznaki przetrenowania.

## Podsumowanie

Model wciąż nie można uznać za dobry w kontekście problemu klasyfikacji win, jednak okazał się od lepszym od modelu benchmarkowego, Bayes Naive Classifier.

## 4. Model 3 - Random Forest

Cechy modelu:

- Może radzić sobie z dużą liczbą zmiennych i skutecznie przetwarzać zmienne kategoryczne.
- Model oferuje możliwość oceny ważności każdej cechy w procesie decyzyjnym, co może być użyteczne w analizie danych.
- Random Forest zazwyczaj oferuje lepszą dokładność niż pojedyncze drzewa decyzyjne, ponieważ agregując wyniki wielu drzew redukuje ryzyko nadmiernego dopasowania
- Łatwy w interpretacji

Kroki wykonane w ramach tworzenia modelu:

1. Inicjalizacja danych za pomocą StandardScalera.
2. Podział danych na zestaw treningowy i testowy w proporcji 4:1, co pozostawiło nas z 1202 rekordami z zbiorze treningowym i 301 w zbiorze testowym.
3. Wybranie strategii próbkowania.
4. Utworzenie obiektu klasy RandomForestClassifier.
5. Uczenie i ewaluacja modelu o standardowych hiperparametrach przy wykorzystaniu walidacji krzyżowej (do celów poglądowych).
6. Definicja zbioru hiperparametów do zoptymalizowania.
7. Wyznaczenie optymalnych hiperparametrów.
8. Uczenie i ewaluacja modelu o standardowych hiperparametrach przy wykorzystaniu walidacji krzyżowej.
9. Wytrenowanie modelu o zoptymalizowanych parametrach.
10. Wykonanie przez model predykcji na zbiorze testowym.

```
In [ ]: rapport_functions.random_forest_main()
```

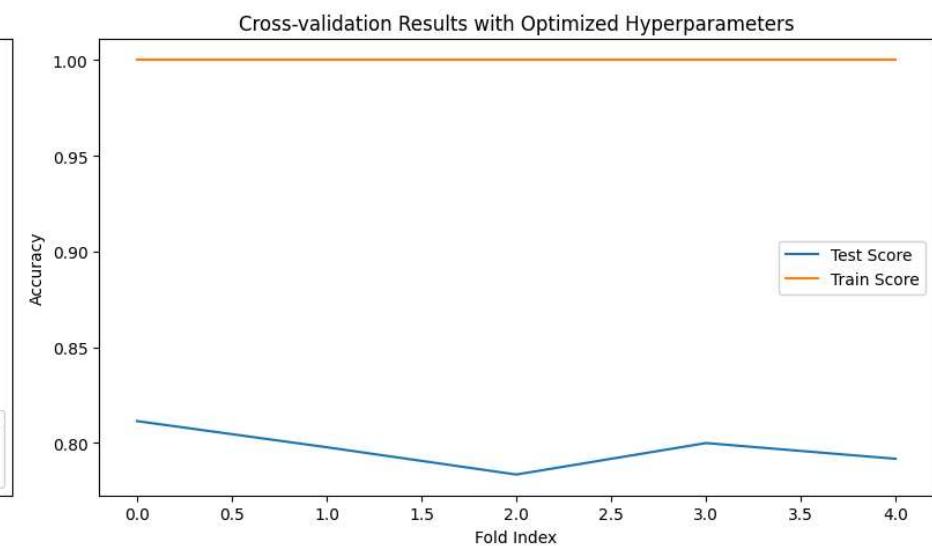
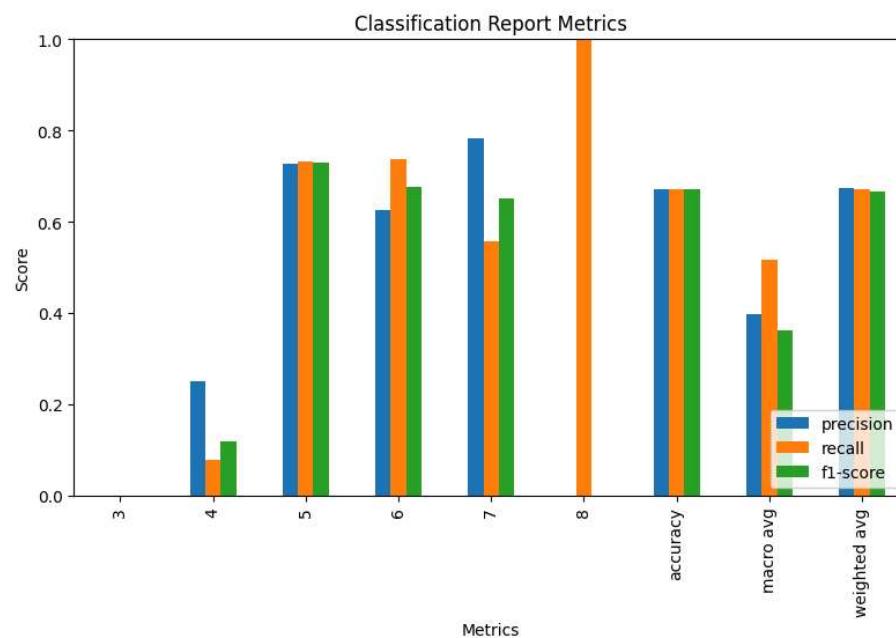
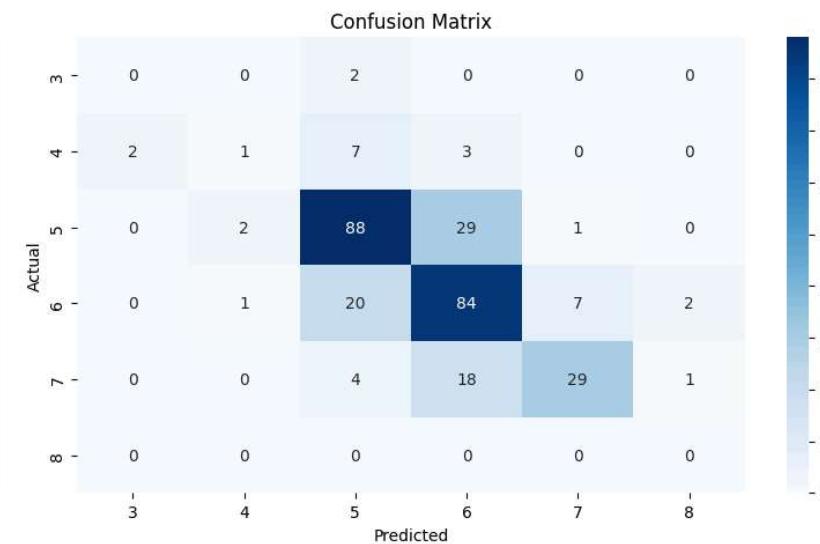
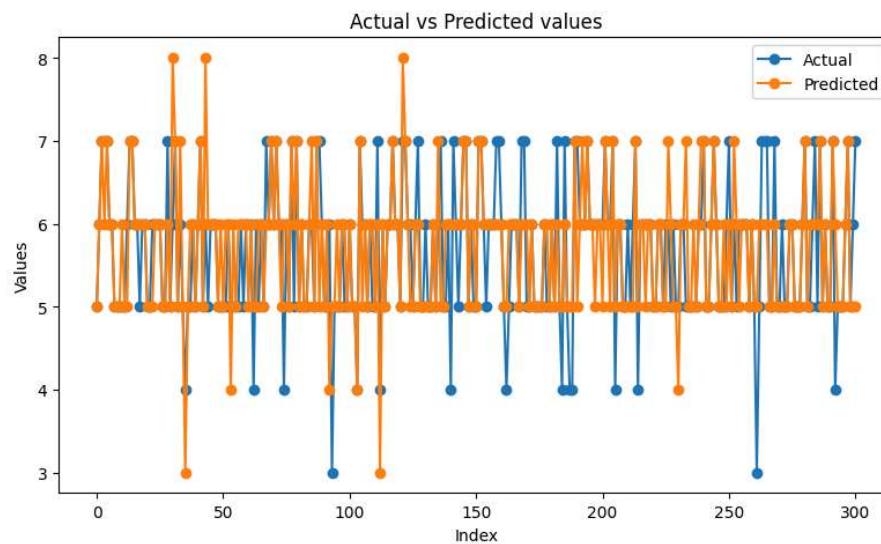
```
2024-06-06 00:11:21,149 - INFO - Data initialization and splitting complete.  
Fitting 5 folds for each of 50 candidates, totalling 250 fits
```

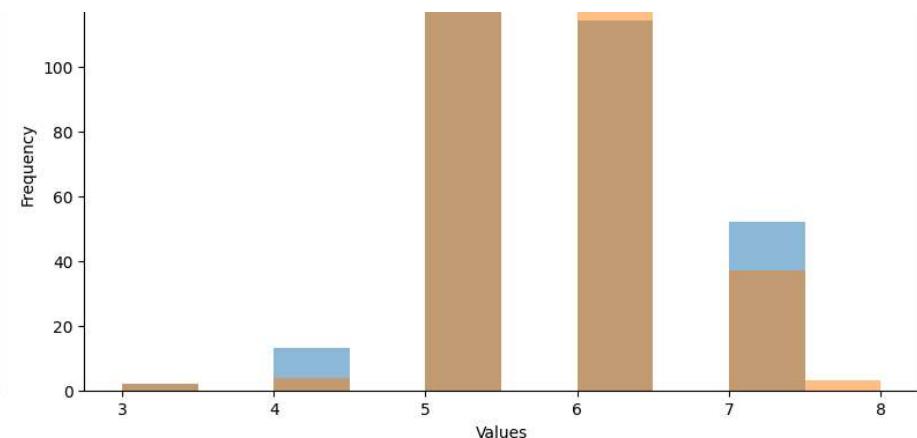
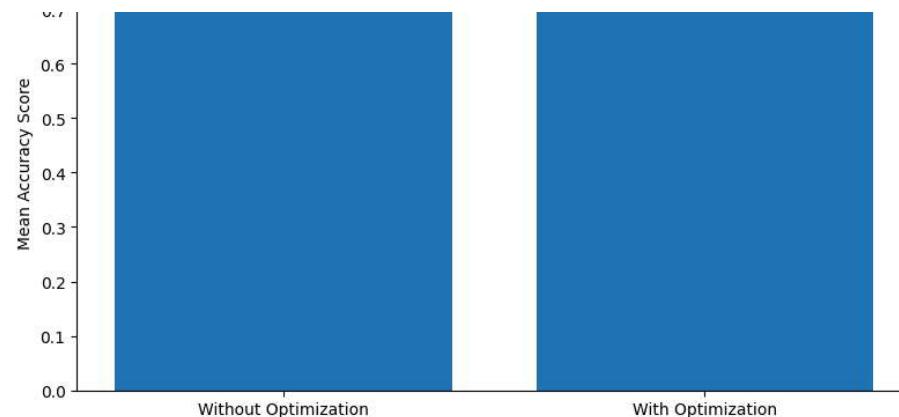
```
2024-06-06 00:12:26,145 - INFO - Best parameters: {'n_estimators': 300, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 50, 'bootstrap': True}
```

```
2024-06-06 00:12:33,225 - INFO - Classification report:
```

	precision	recall	f1-score	support
3	0.00	0.00	0.00	2
4	0.25	0.08	0.12	13
5	0.73	0.73	0.73	120
6	0.63	0.74	0.68	114
7	0.78	0.56	0.65	52
8	0.00	1.00	0.00	0
accuracy			0.67	301
macro avg	0.40	0.52	0.36	301
weighted avg	0.67	0.67	0.67	301

0.6710963455149501





## Ocena modelu:

### Optymalne hiperparametry

Za optymalne hiperparametry spośród podanych do wyznaczenia uznano:

- n\_estimators: 300,
- min\_samples\_split: 2,
- min\_samples\_leaf: 1,
- max\_features: sqrt,
- max\_depth: 50,
- bootstrap: False.

## Wyniki modelu

- model osiągnął dokładność na poziomie 67%,
- klasą o najwyższym poziomie F1 była klasa 5, miara ta wyniosła dla niej 0.75. Oprócz niej, pułap 0.60 osiągnęły klasy 6 i 7, z wynikami kolejno 0.67 i 0.64,
- miał tendencję do zbyt częstego w stosunku do danych rzeczywistych przydzielania win do klasy 6, za to klasa 7 cierpała na niedobór zaklasyfikowanych win w stosunku do stanu rzeczywistego,
- wraz ze spadkiem liczby przedstawicieli danej klasy jakościowej w zbiorze danych, spadała także skuteczność modelu w klasyfikowaniu win do tychże klas,

- model okazał wyraźnie oznaki przetrenowania.

## Podsumowanie

Model lasu losowego okazał się najlepszym z rozpatrywanych. O ile nadal pozostawia on wiele do życzenia, ze względu na względnie zadowalające wyniki, jego użycie ma potencjał przynieść wartość.