Raport programu badającego diagnozowanie raka piersi

Dominika Wojtanowska, Piotr Dębowski

January 2021

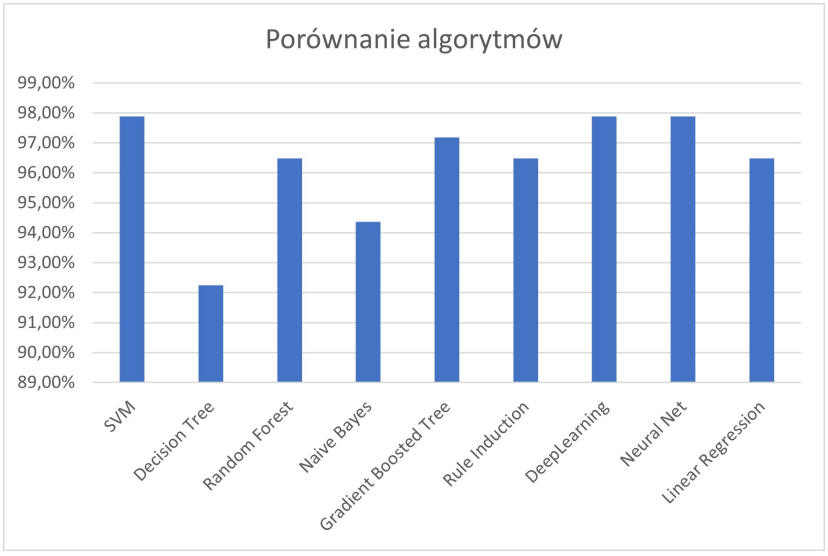
1 Wstęp

Dany projekt przedstawi program wykrywający raka piersi na podstawie parametrów wyciągniętych ze zdjęcia (między innymi promień, tekstura, symetria itp.). Wszystkie parametry są liczbami rzeczywistymi i jest ich w sumie 30. Zbiór danych zawiera 32 atrybuty z czego, gdzie kolumny 3-32 to wspomniane wcześniej parametry ze zdjęć. Atrybutem decyzyjnym jest kolumna druga z dwiema możliwymi odpowiedziami M = malignant (złośliwy), B = benignn (łagodny), natomiast kolumna pierwsza zawiera numer ID badania. Danych jest 569.

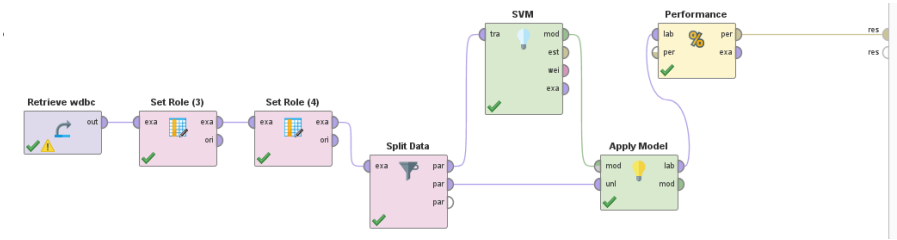
Rysunek 1: Przykładowe zdjęcie mammograficzne

2 Analiza i dobranie algorytmu

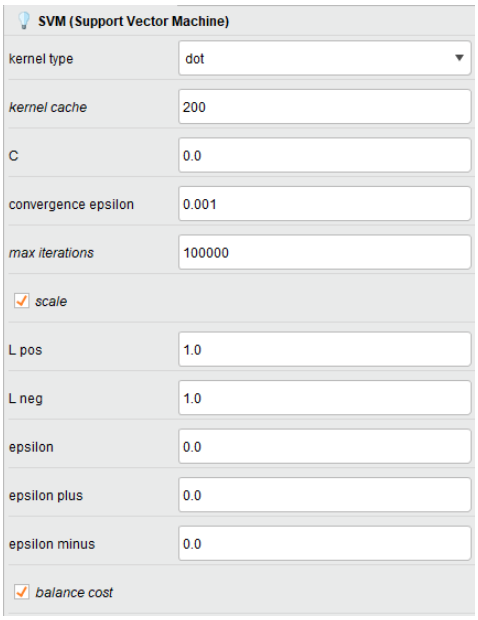
By dobrać odpowiedni algorytm skorzystaliśmy z Rapid Minera. Dane były już wyczyszczone i nie posiadały brakujących wartości. Przetestowaliśmy je dla najistotniejszych modeli. Z wyników wyszło iż sieci neuronowe i SVM najlepiej radziły sobie z wykrywaniem raka na danych treningowych bo aż z dokładnością 97,89. Drzewo decyzyjne CART również znacznie nie odstawało w swojej precyzji (96,48). Klasyfikator Bayesowski osiągnął również precyzje powyżej 90. Na podstawie tej analizy wybraliśmy wykorzystanie SVM w naszym programie.



Rysunek 2: Porównanie algorytmów



Rysunek 3: Proces z programu Rapid Miner do testu SVM



Rysunek 4: Parametry SVM zastosowane w modelu

accuracy: 97.89%

	true M	true B	class precision
pred. M	50	0	100.00%
pred. B	3	89	96.74%
class recall	94.34%	100.00%	

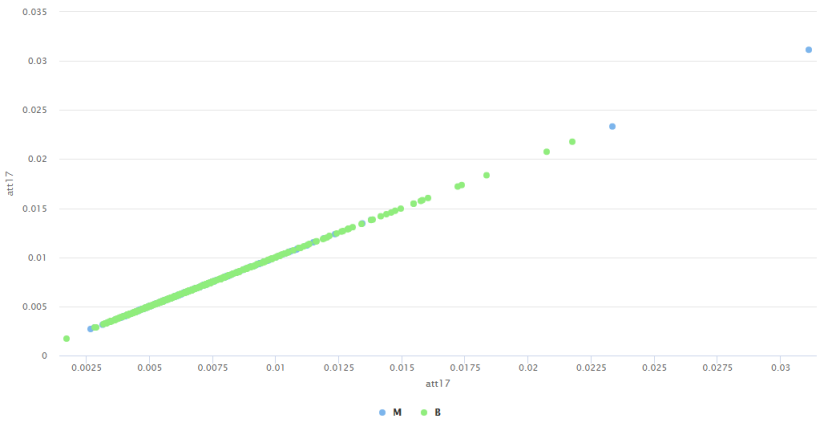
Rysunek 5: Precyzja algorytmu SVM

3 Preprocessing danych

Dane były w stanie dość „czystym", więc nie potrzebowały dużej obróbki. Jako, że były to atrybuty numeryczne, dokonano na nich normalizacji, która nieznacznie poprawiła predykcję. Nie zaobserwowaliśmy również znacznie odstających wyników. Pozbyto się kolumn nieznaczących, takich jak na przykład atrybut 17. Kolumny te wyznaczono posiłkując się wykresami. Przy tak małej liczbie cech była to optymalna opcja.

Wykorzystując metodę „stratify" podzieliliśmy dane tak, że 75% z nich trafiło do zbioru uczącego, a pozostałe 25% do zbioru testowego.

```
1 data = pandas.read_table("wdbc.data", sep=",", header=None)
2 data.drop([data.columns[13], data.columns[10], data.columns[11],
  ↪  data.columns[16], data.columns[31]], axis=1).head(2)
3
4 y = data.iloc[:, 1].values
```



Rysunek 6: Wykres atrybutu 17

```
train, test = train_test_split(data, test_size = 0.25, random_state = 1,
↪  stratify=y)
```

4 Budowa modelu

Do budowy naszego klasyfikatora wykorzystaliśmy język Python i bibliotekę sklearn. W celu znalezienia optymalnego rozwiązania przeprowadziliśmy szereg eksperymentów, w celu dobrania najlepszych parametrów.
Eksperyment z karnelem wielomiannowym stopnia trzeciego i parametrem regularayzacji $C = 13$

```
classifierPoly = Pipeline([
    ("scaler", StandardScaler()), ("svm,clf",SVC(kernel="poly", C=13))])
classifierPoly.fit(x, y)
Y_pred = classifierPoly.predict(X_test)
cm = confusion_matrix(Y_test,Y_pred)
accuracy = float(cm.diagonal().sum())/len(Y_test)
print("\nAccuracy Of SVM For The Given Dataset karnel=poly: ", accuracy)
```

Accuracy Of SVM For The Given Dataset karnel=poly: 0.958
Eksperyment z karnelem liniowym i parametrem regularayzacji $C = 5$

```
classifier2 = Pipeline([
    ("scaler", StandardScaler()), ("linear_svc", LinearSVC(C=5, loss="hinge"))])
classifier2.fit(x, y)
Y_pred = classifier2.predict(X_test)
cm = confusion_matrix(Y_test,Y_pred)
accuracy = float(cm.diagonal().sum())/len(Y_test)
print("\nAccuracy Of LinearSVC(C=5, loss=hinge): ", accuracy)
```

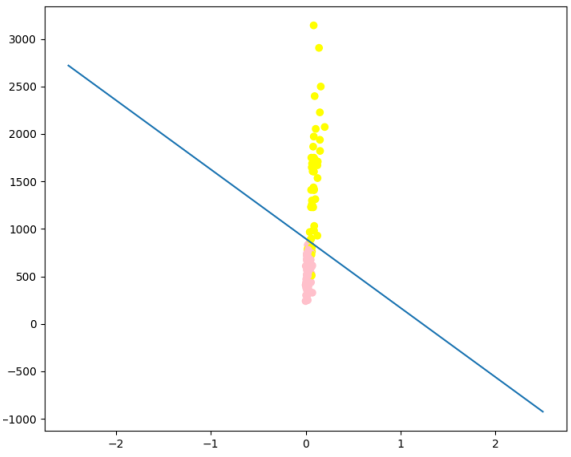
Accurfacy Of LinearSVC(C=5, loss=hinge): 0.958
W wyniku przeprowadzonych przez nas eksperymentów znaleźliśmy następujący optymalny model:

```
classifier = SVC(kernel='linear', random_state = 1, C=10)
classifier.fit(x, y)
Y_pred = classifier.predict(X_test)
cm = confusion_matrix(Y_test,Y_pred)
accuracy = float(cm.diagonal().sum())/len(Y_test)
print("\nAccuracy Of SVM with C=10 For The Given Dataset : ", accuracy)
```

Accuracy Of SVM with C=10 For The Given Dataset : 0.965

5 Wizualizacja modelu dla dwóch istotnych parametrów

By łatwiej było przedstawić idee maszyny wektorów nośnych, wybraliśmy 2 znaczące atrybuty. Wiadomo, iż precyzja predykcji była znacząco mniejsza, gdyż wyniosła 92%. Mimo to wyodrębnienie tych dwóch atrybutów pozwoliło nam naszkicować wykres predykcji zbioru testowego.



Rysunek 7: Wykres SVM

```
color = ["pink" if c=='B' else "yellow" for c in Y_test]
plt.scatter(X_test[:, 0], X_test[:, 1], c=color)
w = svc.coef_[0]
a = -w[0] / w[1]
xx = np.linspace(-2.5 , 2.5)
yy = a * xx - (svc.intercept_[0]) / w[1]
plt.plot(xx,yy)
plt.show()
```

Literatura

„Uczenie maszynowe w Pythonie. Receptury” Chris Albon