Uniwersytet Jagielloński w Krakowie

Wydział Fizyki, Astronomii i Informatyki Stosowanej

Pavlo Boidachenko

Nr albumu: 1124969

Aplikacja uczenia maszynowego metodą SVM

Praca licencjacka
na kierunku informatyki

Praca wykonana pod kierunkiem
dr Grzegorz Surówka
Zakład Technologii Informatycznych

Kraków 2019

Oświadczenie autora pracy

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

.....	.....
Kraków, dnia	Podpis autora pracy

Oświadczenie kierującego pracą

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

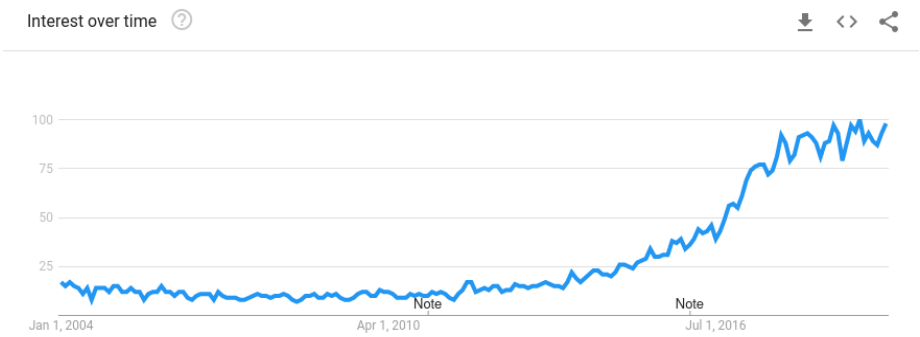
.....	.....
Kraków, dnia	Podpis kierującego pracą

Spis treści

1 Wstęp

1.1 Motywacja

W aktualne czasy temat Uczenia Maszynowego jest popularny1.1 jak nigdy do tego. Projekty z użyciem Uczenia Maszynowego pozwalają na tworzenie aplikacji które jeszcze 10 lat temu trudno było wyobrazić.



Rysunek 1.1: Machine Learning trends
Źródło: Google Trends

Rozpowszechnienie Uczenia Maszynowego również spowodowało i moje zainteresowanie tematem. Z tego powodu dla swojej pracy licencjackiej wybrałem temat: Aplikacja uczenia maszynowego metodą SVM. Po zakończeniu pracy spodziewam się podwyższyć swoją kompetencje w dziale Uczenia Maszynowego.

1.2 Cel

Celem mojej pracy licencjackiej jest stworzenie oprogramowania pozwalającego na generowanie modeli używając Maszyny wektorów wspierających *(ang. Support Vector Machine, SVM)* z graficznym interfejsem użytkownika. Program będą mogli użyć osoby potrzebujące szybko przetrenować kilka modeli, przetestować ich dla różnych parametrów, zwizualizować dane. Program ma na celu ułatwienie pracę z Maszyną wektorów wspierających poprzez graficzny interfejs użytkownika oparty na bibliotekę QT. Część funkcjonalna programu jest oparta o bibliotekę LIBSVM[**CC01a**].

1.3 Zakres

Program powinien móc ustawiać parametry dla wybranej metody oraz jądra *(*ang. kernel), generować wykresy podawanych zbiorów danych, interpretować różne formaty zbiorów danych, wykonywać Sprawdzian krzyżowy (ang. Cross validation, CV), mieć metodę do optymalizacji parametrów, pokazywać wyniki trenowania oraz testowania modeli.

2 Metoda klasyfikacji SVM

W tym paragrafie w ogólnych zarysach jest opisana Maszyna Wektorów Wspierających oraz jej typy zaimplementowane w LIBSVM. Również będą krótkie opisy zaimplementowanych jąder.

2.1 Opis

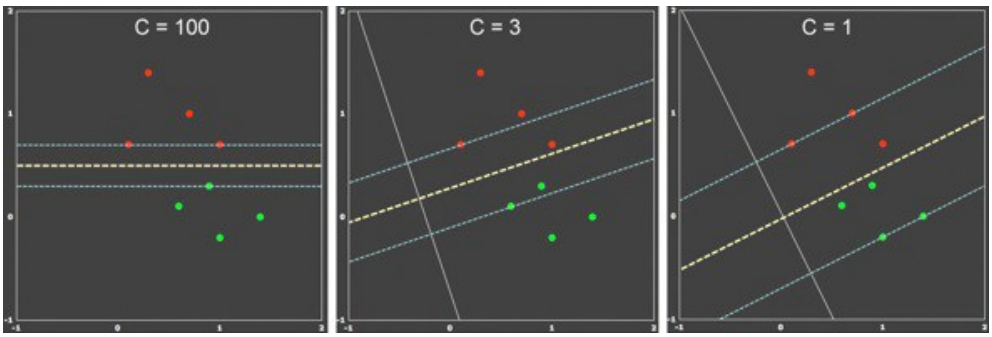
Swój program napisałem w oparciu o bibliotekę LIBSVM[**CC01a**]. Maszyna Wektorów Wspierajacych(ang. Support Vector Machine. SVM) - klasyfikator, nauka którego ma na celu wyznaczenie hiperpłaszczyzny rozdzielającej dwie klasy z maksymalnym marginesem. Zaletą takiego klasyfikatora jest to że po uczeniu margines mówi jak dobrze są odseparowane klasy. LIBSVM implementuje pięć typów Maszyny Wektorów Wspierajacych C-SVC, ν-SVC, One class SVM, ϵ-SVR, ν-SVR.

2.2 C-SVC

C-Support Vector Classification - rodzaj klasyfikatora używający C jako parametr regularyzacji. Jeśli jest dany wektor $x_i \in R^n$, $i = 1, ..., l$ w dwóch klasach i wektor etykiet $y_i \in \{1, -1\}$ to C-SVC rozwiązuję tak sformułowany problem:

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\varepsilon} \quad & \tfrac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\sum_{i=1}^{l}\varepsilon_i \\ \text{Z zastrzeżeniem że} \quad & y_i(\boldsymbol{\omega}^T\phi(x_i) + b) \geqslant 1 - \varepsilon_i \\ & \varepsilon_i \geqslant 0, i = 1, ..., l \end{aligned}$$

Parametr C służy do ustawienia marginesu: duży C $\rightarrow$ mały margines, mały C $\rightarrow$ duży margines.



Rysunek 2.1: Zależność marginesu od parametru C
Źródło: `https://medium.com/@pushkarmandot`

Dobry model dobrze separuję dane i razem z tym ma duży margines. Natomiast w w rzeczywistości jedno wyłącza drugie: duży margines włącza punkty z dwóch klas, a dobre separowanie może powodować przeuczanie(ang. Overfitting). Przeuczanie może skutkować tym że model jest dobry na danych treningowych ale jest zły na danych testowych.

2.3 ν-SVM

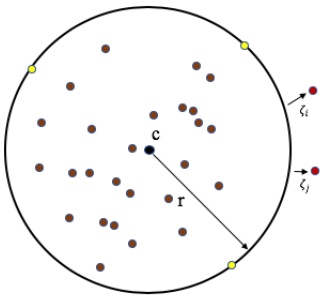
ν-Support Vector Classification - rodzaj klasyfikatora używający ν jako parametr regularyzacji. Jest bardzo podobny do C-SVM, z różnicą że $\nu \in [0, 1]$. Przyjemną właściwością ν jest to że on jest dolną granicą stosunku wektorów wspierających i górną granicą stosunku błędu uczenia.

Jeśli jest dany wektor $x_i \in R^n$, $i = 1, ..., l$ w dwóch klasach i wektor $y \in R^l$ taki że $y_i \in \{1, -1\}$ to pierwotny problem optymalizacji wygląda następująco:

$$\begin{aligned} \min_{\boldsymbol{\omega}, b, \varepsilon, \rho} \quad & \tfrac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} - \nu\rho + \tfrac{1}{l}\sum_{i=1}^{l}\varepsilon_i \\ \text{Z zastrzeżeniem że} \quad & y_i(\boldsymbol{\omega}^T\phi(x_i) + b) \geqslant \rho - \varepsilon_i \\ & \varepsilon_i \geqslant 0,\ i = 1, ..., l,\ \rho \geqslant 0 \end{aligned}$$

2.4 One-class SVM

One-class Support Vector Machine - rodzaj klasyfikatora uczenia nienadzorowanego, które zakłada brak etykiet w danych uczących. Ma na celu znalezienie niewiadomych wzorców/anomalii(klastrów) w danych wejściowych.



Rysunek 2.2: Hipersfera zawierająca punkty danych. Ma środek c i promień R. Punkty na krawędzi są wektorami wspierającymi.
Źródło: Wikipedia

Jeśli dany jest wektor $x_i \in R^n$, $i = 1, ..., l$ bez informacji o klasach, to pierwotny problem optymalizacji wygląda następująco:

$$\begin{aligned} \min_{\boldsymbol{\omega}, \varepsilon, \rho} \quad & \tfrac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} - \rho + \tfrac{1}{\nu l}\sum_{i=1}^{l}\varepsilon_i \\ \text{Z zastrzeżeniem że} \quad & \boldsymbol{\omega}^T\phi(x_i) \geqslant \rho - \varepsilon_i, \\ & \varepsilon \geqslant 0,\ i = 1, ..., l \end{aligned}$$

2.5 ϵ-SVR

Jeśli wektor etykiet $y_i \in R$ to jest używana metoda regresji. ϵ-Support Vector Classification - używa C i ϵ jako parametrów regularyzacji. Celem jest znalezienie takiej funkcji $f(x)$ że jej wartość odchyla się od y_n na wartość nie większą od ϵ dla każdego punktu z zbioru treningowego.

Jeśli jest dany zbiór danych treningowych $\{(x_1, x_1), ..., (x_l, z_l)\}$, gdzie $x - I \in R^n$ jest wektorem cech, a $z_i \in R^1$ jest wyjściem. Przy danych parametrach $C > 0$ i $\epsilon > 0$, standardowa forma SVR to:

$$\begin{aligned} \min_{\boldsymbol{\omega}, b, \boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}^*} \quad & \tfrac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\sum_{i=1}^{l}\varepsilon_i + C\sum_{i=1}^{l}\varepsilon_i^* \\ \text{Z zastrzeżeniem że} \quad & \boldsymbol{\omega}^T\phi(x_i) + b - z_i \leqslant \epsilon + \varepsilon_i, \\ & z_i - \boldsymbol{\omega}^T\phi(x_i) - b \leqslant \epsilon + \varepsilon_i^*, \\ & \varepsilon_i, \varepsilon_i^* \geqslant 0, i = 1, ..., l \end{aligned}$$

2.6 ν-SVR

ν-Support Vector Regression - podobnie do ν-SVC, używa parameter $\nu \in (0, 1]$ dla kontroli liczby wektorów wspierających. Również używa parametru ϵ. Z parametrami (C, ν) ν-SVR rozwiązuję:

$$\begin{aligned} \min_{\boldsymbol{\omega}, b, \boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}^*, \epsilon} \quad & \tfrac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C(\nu\epsilon + \tfrac{1}{l}\sum_{i=1}^{l}(\varepsilon_i + \varepsilon_i^*)) \\ \text{Z zastrzeżeniem że} \quad & (\boldsymbol{\omega}^T\phi(x_i) + b) - z_i \leqslant \epsilon + \varepsilon_i, \\ & z_i - (\boldsymbol{\omega}^T\phi(x_i) + b) \leqslant \epsilon + \varepsilon_i^*, \\ & \varepsilon_i, \varepsilon_i^* \geqslant 0,\ i = 1, ..., l,\ \epsilon \geqslant 0 \end{aligned}$$

2.7 Jądra

3 Projekt aplikacji

3.1 Opis

3.2 Technologie

4 Podsumowanie

4.1 Odniesienie do celu pracy

4.2 Co można dodać