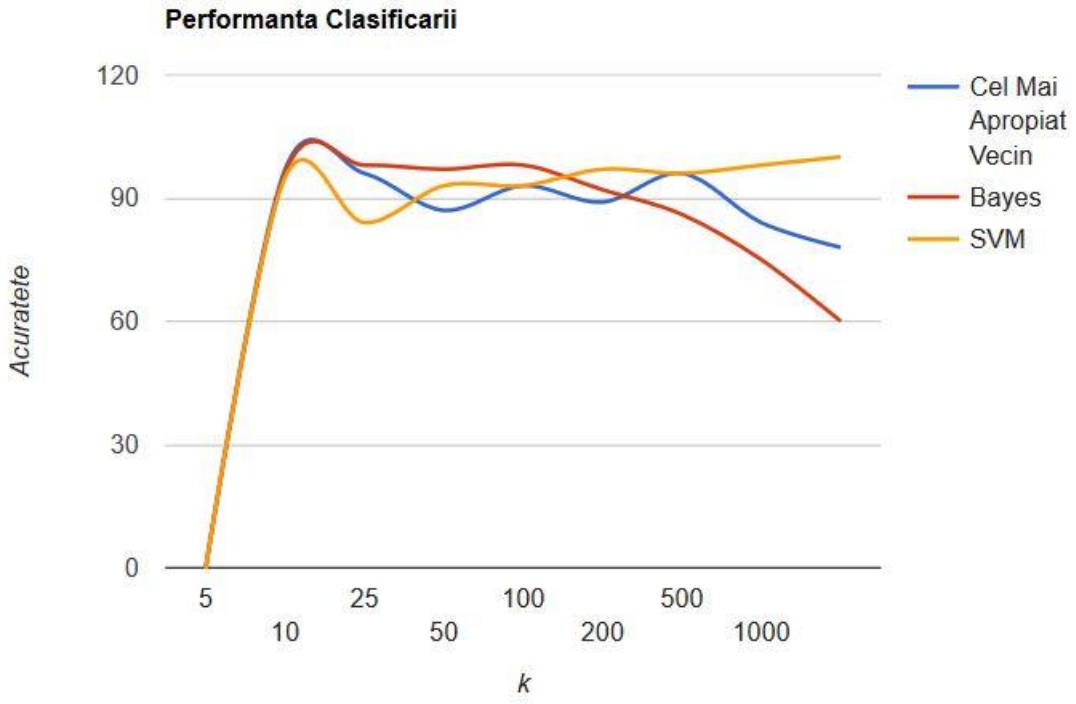
Monica Pascu
Grupa 351

Proiect 3

Clasificarea imaginilor folosind modelul Bag-Of-Visual-Words

a)



Cel mai bun clasificator din punct de vedere al performanței de clasificare este SVM. SVM încearcă să găsească un hiper-plan, separând diferitele clase ale mulțimii de antrenare, cu o marjă mare între punctele de date și limita de decizie. Practic, încearcă să creeze un gard între cele 2 clase asftel încat cât mai puține exemple să fie pe partea greșită, iar distanța dintre cele 2 părți să fie cât mai mare.

Cele mai importante exemple din mulțimea de antrenare sunt cele care alcătuiesc granița. Efectuarea compromisului între numărul de cazuri pe care trebuie să le permită pe partea greșită și cât de complexă este limita dintre cele 2 clase poartă numele de model. Modelele acestea sunt, în general, mult mai complexe decât clasificatorii bazați pe distanțe, dar iau în considerare mai multe informații atunci când clasifică datele de test și vor avea în general o precizie mult mai bună.

Cel mai rapid este Bayes. Acesta necesită doar media deviației standard din clasele datelor. Numai probabilitățile se calculează în timpul clasificării, astfel încât performanța sa nu este afectată de mărimea mulțimii de antrenare.

b) K optim pentru Cel mai apropiat vecin : între 5 și 10. 5 dacă trebuie să aleg.

K optim pentru Bayes : între 5 și 25.

K optim pentru SVM : 500.

O solutie ar fi sa selectam caracteristicile care prezic cel mai bine calitatea de membru al unei clase.

Folosirea unei metode de filtrare a caracteristicilor separate pentru a identifica caracteristicile informative, pentru a minimiza tendința de prejudecata(bias-ul), ar fi o idee. (Selectarea recurentă a caracteristicilor folosind clasificatorul este un mod mai părtinitor de alegere decât filtrarea.)

Separarea procesului de filtrare a caracteristicilor de etapa de clasificare ar putea, de asemenea, să se dovedească a fi benefică pentru rezultatele extrase din datele viitoare (care nu fac parte din batch-ul de testare sau antrenare).

c) Cu un număr de vecini de 5 (sau mai mic), separarea dintre Mașini și nonMașini este foarte zimțată. În plus, există "insule" ale nonMașinilor pe teritoriul Mașinilor și invers. Pe măsură ce numărul de vecini crește până la, de exemplu, 20, tranziția devine mai ușoară, iar insulele dispar și împărțirea dintre Mașini și nonMașini face o treabă bună de a urma linia de graniță. Cand numărul devine foarte mare, să zicem, 80, distincția dintre cele două categorii devine mai estompată, iar linia de predicție a limitei nu se potrivește foarte bine.

La numere mai mici (cum e 5) locațiile insulelor și curbele exacte ale granițelor se vor schimba radical pe măsură ce se colectează date noi.

Unele imagini de test au fost clasificate greșit din cauza ”insulelor” menționate anterior. Dacă cei mai apropiați vecini ai unei Mașini sunt, să zicem, 3 nonMașini și 2 Mașini, algoritmul o să clasifice Mașina greșit.

Orice miscare random a setului de date de antrenare ar putea schimba dramatic modelul. Daca am alege 10 vecini, in loc de 5, algoritmul ar fi mai robust, dar în același timp mai rigid. Alegerea numarului de vecini depinde, în final, de setul de date.

d) Algoritmul alege cele k caracteristici (centrii clusterilor/cuvintele vizuale) într-un mod random, de aceea n-o să avem constant aceleași caracteristici k la rulări repetate. Ideea este de a păstra caracteristica cea mai relevantă, dar nu redundantă pentru modelul predictiv, care poate oferi o precizie optimă. Alegand întotdeauna random, nu putem păstra asta, iar precizia nu mai este optimă.

De asemenea, selecția caracteristicilor nu are în mod necesar o precizie optimă atunci când caracteristicile sunt interdependente și nu se exclud reciproc. (putem să avem același patch de șosea atât la o imagine Mașină, cât și la una nonMașină)

Având în vedere cele menționate anterior, nu putem să ne așteptăm la o performanță foarte ridicată (aproape de 100%) la fiecare rulare, din cauza modului random de selecție al caracteristicilor.