Dokumentácia štvrtého zadania z UI

Adam Štuller

May 6, 2019

1 Riešený problém – Zadanie

Zadaním bolo implementácia rozpoznávanie vzorov pomocou strojového učenia. Konkrétne išlo o rozpoznávanie ručne písaných číslic z MNIST datasetu. Išlo o klasifikačný problém, kde sme pole pixelov na vstupe (každý pixel bol reprezentovaný číslicou od 0 do 255) pomocou troch algoritmov strojového učenia klasifikovali do jednej z desiatich tried, teda ako číslicu od 0 do 9.

V datasete, ktorý sme mali k dispozícií bolo 60 000 príkladov na trénovanie a ďalších 10 000 na testovanie. Každý z príkladov bol olabelovaný a teda sa jednalo o príklad supervised machine learning. Tri algoritmy, ktoré som použil boli Neurónová sieť, Rozhodovací strom a Náhodný les. Vo všetkých prípadoch som použil Pythonovskú knižnicu scikit-learn, ktorá všetky tieto modely poskytuje a ponúka veľa možností na prácu s nimi.

Natrénované modeli som potom ukladal pomocou joblib kniznice.

2 Spracovanie dát

Dáta boli uložené vo formáte csv. Spracovával som ich pomocou knižnice pandas. Stiahnuté dáta boli dopredu rozdelené na trénovaciu a testovaciu množinu a preto som ich nemusel ďalej upravovať. Stačilo ich rozdeliť na vstupné atribúty a labele. Vstupných atribútov bolo pôvodne 28x28 teda 784. Neskôr som pridal štyri vlastné.

Data som pred použitím normalizoval, teda z čísel od 0 do 255 som ich pretransformoval na čísla v rozmedzí o 0 do 1. Použil som na to dunkciu normalize z kniznice scikit-lear.

3 Neurónová sieť

Prvým použitým algoritmom bola neurónová sieť. Použil som na ňu MLPClassifier, čo je obyčajná neurónová sieť. Natrénoval som ju najprv na trénovacej množine potom som ju testoval jednoducho cez testovaciu množinu ale aj pomocou cross validation.

3.1 Úspešnosť

Jej úspešnosť po trénovaní bola na testovacej množine 97.5% a chybovosť teda 2.5%.

Na obrázku 1 je vidno krivka učenia sa neurónovej siete, teda postup ako sa zlepšovala jej úspešnosť s množstvom dát na ktorých sa učila.

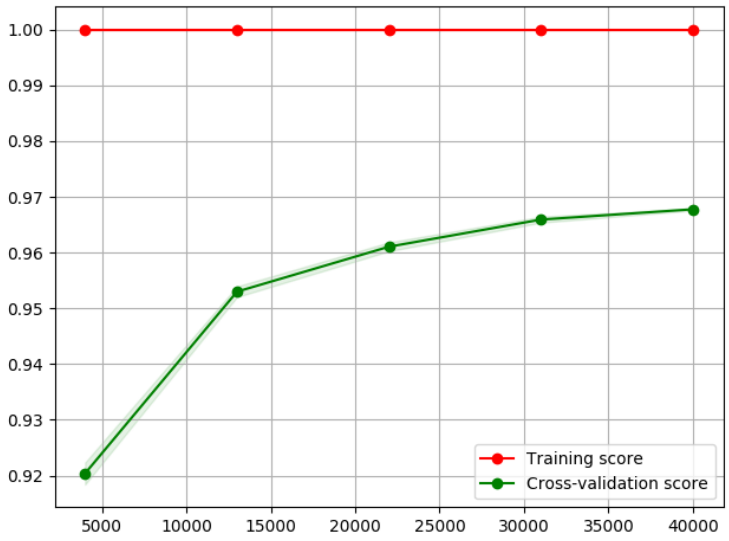

Figure 1: Krivka učenia sa

Na tabulke 1 je vidno percentuálnu úspešnosť neurónovej siete pri rozoznávaní jednotlivých číslic.

Label	Accuracy
0	0.988776
1	0.992070
2	0.974806
3	0.968317
4	0.978615
5	0.964126
6	0.977035
7	0.975681
8	0.960986
9	0.966303

Table 1: Table of accuracy of nerural network fo each individual digit

3.2 Confusion matrix

Obrázok 2 zobrazuje confusion matrix, vypočítanú na základe výsledkov testovacej množiny. Na obrázku je jasne vidno, že aj obyčajná neurónová sieť je v úlohe klasifikátora veľmi dobrá a oproti iným modelom vedie. Najväčší problém mala s odlíšením čísla 4 od 9. V tejto dvojici nesprávne klasifikovala 23 čísel z testovacej množiny.

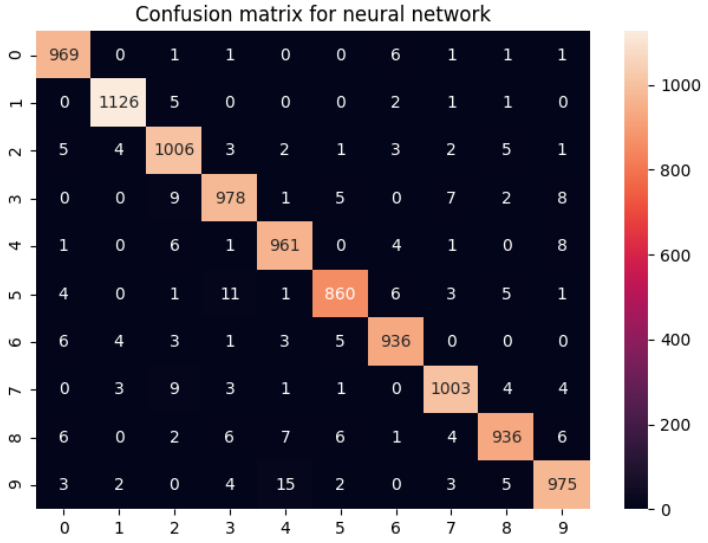

Figure 2: Confusion Matrix

Na obrázku 3 je vidno relatívnu početnosť chýb v klasifikovaní jednotlivých číslic. Je vidno, že napríklad číslica 1 je veľmi špecificka a preto s ňou neurónová sieť nemá takmer žiadne problémy.

3.3 Dôležitosť parametrov

Najprv som sa rozhodol dôležitosť parametrov (jednotlivých pixelov) pri neurónovej sieti nájsť pomocou váh jednotlivých vstupov do siete. Po dlhšom rozmýšľaní som sa ale rozhodol to do projektu nezahrnúť, nakoľko moja neurónová sieť nie je jednovrstvová a váhy parametrov na prvej vrstve neznamená, že ti isté parametre zavážia najviac aj v celkovom výsledku.

Prístup s postupným nulovaním parametrov sa mi tiež nezdal úplne valídny a tak som sa z dôvodu nedostatku vedomostí o problematike rozhodol radšej nevyriekať nepodložené tvrdenia.

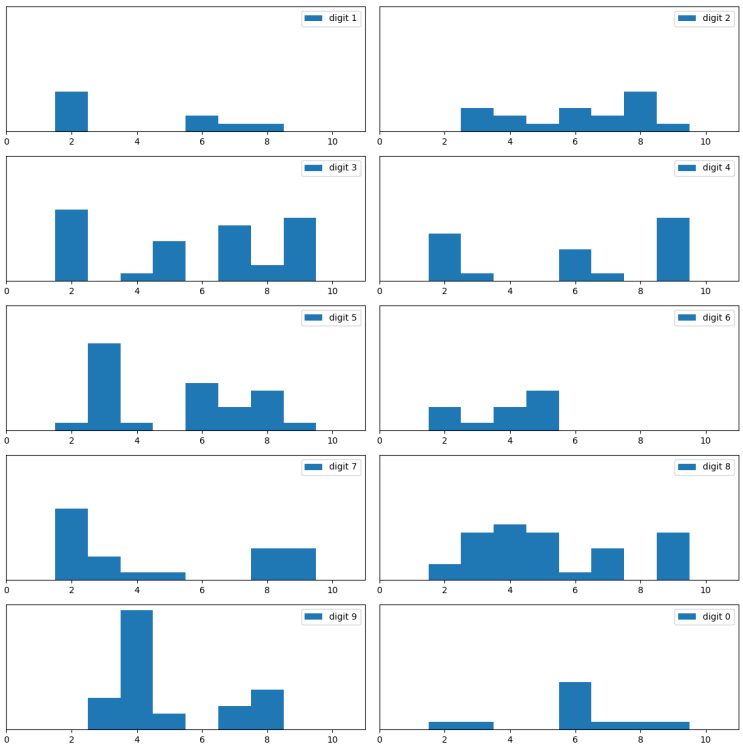

Figure 3: Relatívna početnosť chýb

4 Rozhodovací strom

Rozhodovací strom bol najjednoduchší z použitých algoritmov ale aj najmenej efektívny. Použil som naň DecisionTreeClassifier z scikit-learn. Opäť som ho najprv natrénoval na trénovacej množine a potom otestoval.

4.1 Úspešnosť

Úspešnosť obyčajného rozhodovacieho stromu je oveľa nižšia ako úspešnosť neurónovej siete. Je to 88% a chybovosť je teda 12%. Po cross validácií je priemer úspešnosti ešte menši a síce 0.87 percent. Z podstaty cross validácií sa dá očakávať, že toto je presnejší výsledok.

Graf 4 zobrazuje krivku učenia sa rozhodovacieho stromu s rastúcim množstvom videných príkladov.

4.2 Confusion matrix

Confusion matrix o dosť pestrejšia ako pri neurónovej sieti, konkrétne o 9.5%. Vidno ju na obrázku 5

Figure 4: Krivka učenia sa

4.3 Dôležitosť parametrov

Nakoľko je rozhodovací strom jednoduchšie pochopiteľný model ako neurónová sieť a vieme presne prečo algoritmus tvorby rozhodovacieho stromu vytvoril taký strom aký vytvoril vieme aj povedať, že parametre, ktoré sú v rozhodovacom strome vyššie sú dôležitejšie. Na zaklade tejto vlastnosti som vedel z modelu vytiahnuť všetky parametre aj s ich dôležitosťou a zobraziť ich ako mapu 28x28 pixelov. Týmto spôsobom je jasne vidieť, ktoré pixely sú najdôležitejšie. Táto mapa je na obrazku 6. Čím dôležitejšie pixely tým belšia je farba.

5 Náhodný les

Náhodný les je ďalším implementovaným modelom. Konkrétne je to model RandomForestClassifier. Podstatov náhodného lesa je to, že sám rozhodovací strom často nie je veľmi úspešný. Nahodný les potom nie je vlastne nič iné, iba skupina n rozhodovacích stromov, ktorý klasifikujú na základe spoločného uznesenia. V mojom prípade som použil 100 clasifikátorov.

5.1 Úspešnosť

Úspešnosť náhodného lesa je podstatne vyššia ako úspešnosť osamoteného stromu. 0.9674% je to po obyčajnom otestovaní na testovacej množine a 0.9649 je priemerný výsledok cross validácií.

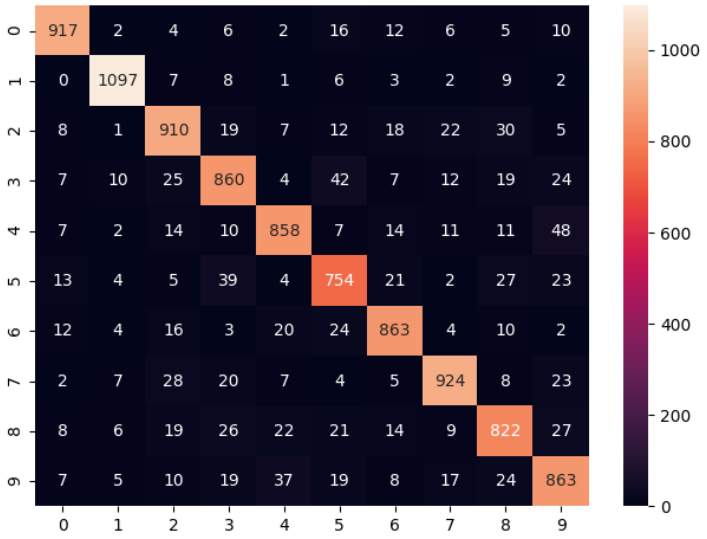

Figure 5: Confusion Matrix Decision Tree

5.2 Confusion matrix

Hodnoty v confusion matrix (obrázok 7) sú opäť veľmi vysoké na diagonále a inde nízke, čo znamená, že je veľmi vysoká úspešnosť. Z confusion matrix je vidno, že klasifikátor ma najväčší problém rozoznať 9 od 4 a 5 od 3.

5.3 Dôležitosť parametrov

Keďže náhodný les je len kombináciou rozhodovacích stromov, dá sa dôležitos't parametrov rozoznať rovnakým spôsobom ako pri rozhodovacích stromoch. Rovnako sa dá aj vizualizovať. Vizualizácia je na obrazku 9.

6 Skombinovanie modelov pomocou VotingClassifier

ďalšou úlohou bolo skombinovať všetky modely do jedného a potom vyhodnotiť ich úspešnosť. Použil som VotingClassifier zo scikit-learn. Najpr som vytvoil všetky tri modeli a potom som ich dal do jedneho konkrétneho, skúsit som soft voting aj hard voting. Porovnanie úspešností samostatných modelov a spojeného modelu je v tabulke 2. Z nej jasne vidno, že spojenie neurónovej siete s rozhodovacím stromom úspešnosti vôbec nepomohlo, ba naopak, zhoršilo ju.

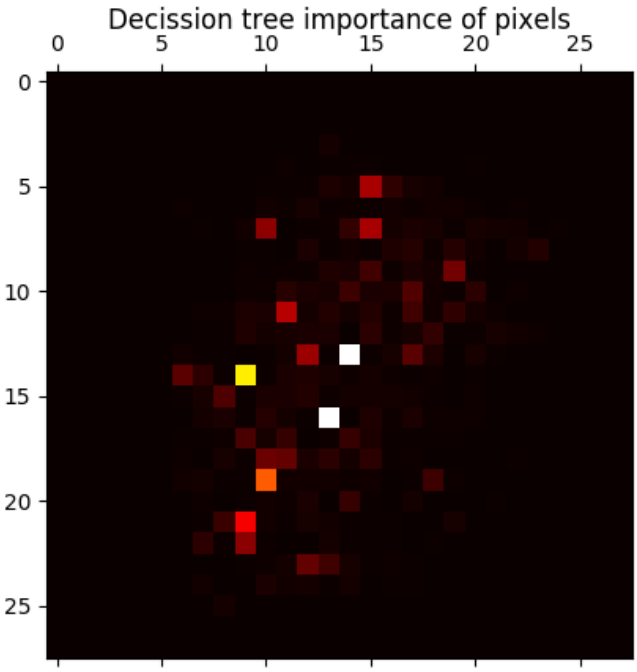

Figure 6: Pixel importance od Decission Tree

Model	Accuracy
Decission Tree	0.88
Random Forest	0.967
Neural Network	0.975
Voting Classifier	0.936

Table 2: Table of accuracy of individual classifiers and combined one

7 Pridanie atribútov

Ďalšou podúlohou v štvrtom zadaní bolo pridať aspoň tri atribúty odvodené z pôvodných vstupov tak aby som celkovo zvýšil úspešnosť klasifikátora aspoň o jedno precento. Táto úloha má ale trosku zádrhelov, nakoľko úspešnosť neurónovej siete je už beztak 97,5 percenta tak je to percento veľmi vysoké číslo.

Nakoniec som množinu vstupných atribútov opravoval dvojakým spôsobom. Najprv som pridal štyri atribúty. Počet pixelov na pravej polovici obrázku, na ľavej, na hornej a na dolnej.

Druhou úpravou bolo odstraňovanie vstupných atribútov, ktoré nemali žiaden

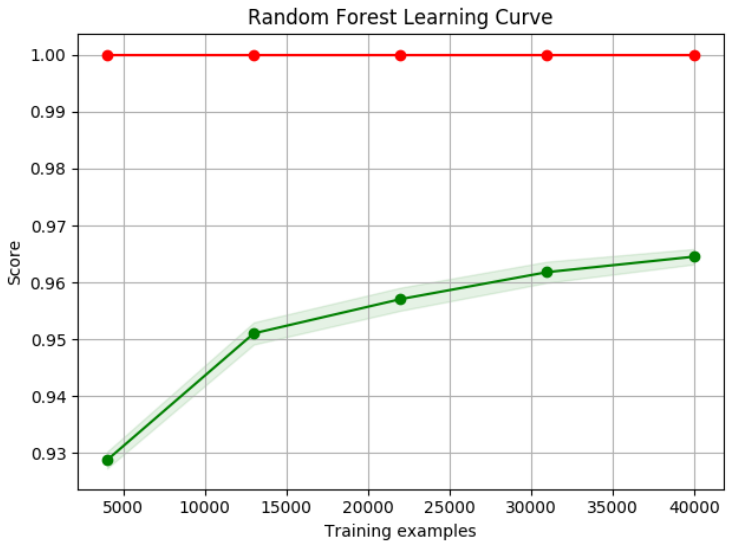

Figure 7: Learning Curve of Random Forest

vplyv na výsledok. Vytvoril som correlation matrix z každého klasifikátora a vstupné atribúty, ktoré nemali žiadnu koreláciu s výstupom som odstránil. Do konfigu ku každému klasifikátorovi som pridal možnosť na úpravu vstupných údajov podla uváženia.

V tabuľke 3 sa nachádzajú úspešnosti klasifikátorov po aplikovaní všetkých druhov úprav.

Model	Original data	Shortened data	Added attributes	Both
Decission Tree	0.886	0.886	0.883	0.886
Random Forest	0.967	0.969	0.969	0.967
Neural Network	0.975	0.976	0.9783	0.975
Voting Classifier	0.936	0.932	0.939	0.938

Table 3: Table of accuracy of classifiers with differently updated dataset

Ako vidno tak pridanie mojich nových parametrov nespôsobilo výrazný nárast v úspešnosti modelov.

8 Aplikácia

Poslednou úlohou (bonusovou) bola tvorba aplikácie na real-time klasifikáciu prísaných čísel. Nakoľko mi však jadro zadania zabralo veľmi veľa času a tvorba podobnej aplikácie by síce nemusela byť zložitá, ale nikdy som nič podobne nerobil, rozhodol som sa, že v záujme svojho času ju neurobím.

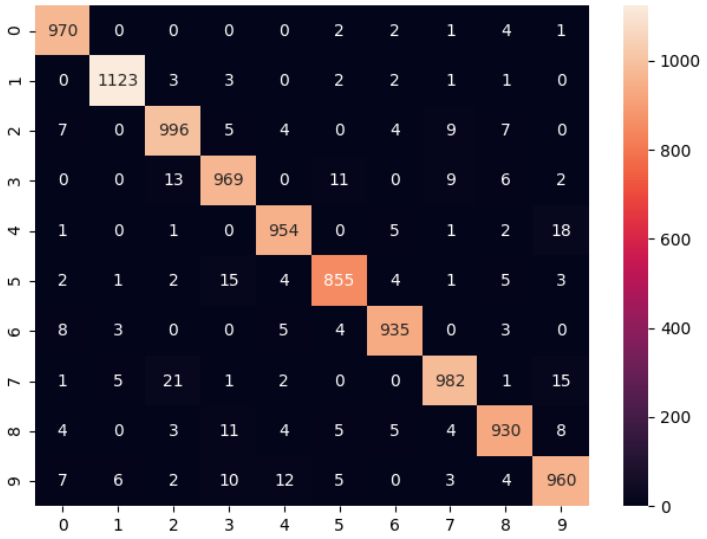

Figure 8: Confusion Matrix Random Forest

9 Záver

Zadanie ma vo všeobecnosti veľmi bavilo ale zabralo mi zo všetkých zadanú z umelej inteligencie najviac času. Pred tým som nemal žiadne skúsenosti, vlastne som ani poriadne nevedel, čo znamená machine learning a nechcel som sa púšťať do používania knižníc bez toho aby som rozumel aspoň čiastočne čo sa deje na pozadí.

Som frustrovaný z toho, že napriek času, ktorý som do toho investoval som nebol schopný sa s knižnicami, ktoré som používal úplne vžiť. Bojovanie s nimi mi zabralo asi najviac času. Ďalšia vec, ktorá ma mrzí je to že som nedokázal poriadne vylepšiť úspešnosť klasfikátorov.

Celkovo som sa ale dozvedel veľmi veľa nových vecí a prvý krát som sa dostal do kontaktu s takým silným nástrojom ako je strojové učenie sa.

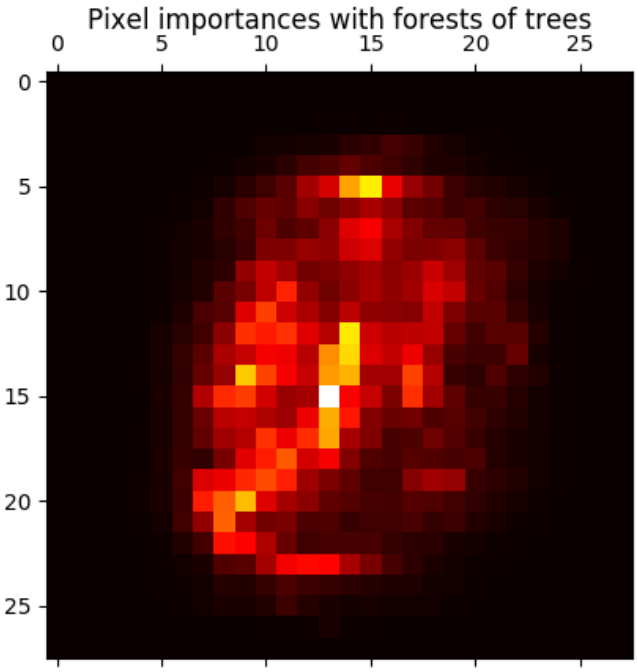

Figure 9: Pixel importance Random Forest