Maskininlärning

Testning med MNIST-datauppsättning

Robert Shaw

EC-Utbildning

ML-Kunskapskontroll2

202403

Abstract

Den här rapporten utforskar skapandet och den lokala distributionen av övervakade maskininlärningsmodeller (ML) för att klassificera handskrivna siffror från MNIST-datauppsättningen (Modified National Institute of Standards and Technology), med betoning på prestanda för stödvektormaskiner (SVM) och slumpmässiga skogar.

Den betonar vikten av utvärderingsmått som noggrannhet, precision, träffsäkerhet och F1-score, och behovet av att upprätthålla klassfördelningen under uppdelningen mellan tåg och test.

Även om Convolutional Neural Networks (CNN) inte var en del av denna studie, är deras potential för överlägsen prestanda inom bildbehandling på grund av djupinlärningsfunktioner erkänd.

Resultaten tyder på att traditionella ML-modeller är mycket lovande, men CNN kan erbjuda förbättrade resultat när det gäller bildklassificering. Denna forskning understryker ML:s viktiga roll för att utveckla bildbehandlingstekniken.

Erkännande

Jag vill uttrycka min uppriktiga tacksamhet till min lärare, Antonio Prgomet, för hans outtröttliga arbete för att hjälpa elever som jag att uppskatta essensen av lärande genom uthållighet genom att arbeta för att förstå de enkla sakerna väl. Bygg sedan långsamt, bryt, bygg och bryt för att bilda ett verkligt djup av förståelse. Hans egna minutiöst förberedda lektioner som flyter så lätt är ett bra exempel på detta, där jag bara kan föreställa mig den ansträngning som krävs bakom kulisserna för att göra datavetenskap lättare att förstå och lära sig.

Jag vill tacka alla mina klasskamrater för att de har hjälpt mig i mina stunder av nöd för att ta reda på vad som inte fungerar och varför. De är en konstant i en stokastisk värld som fungerar på den beprövade premissen av varaktiga lösningar som hittas tillsammans.

Skapas automatiskt i Word genom att gå till Referenser > Innehållsförteckning.

Innehållsförteckning

1 Inledning

Inom det snabbt växande området artificiell intelligens (AI) har utveckling och tillämpning av maskininlärningsmodeller (ML) för bildbehandling blivit ett viktigt forsknings- och innovationsområde. Detta intresse drivs av den ökande efterfrågan inom olika sektorer, inklusive hälso- och sjukvård, där ML kan hjälpa till att diagnostisera sjukdomar från medicinska bilder; inom fordonsindustrin, genom utveckling av autonoma fordon som tolkar visuell information för att navigera säkert; och inom säkerhet, där ansiktsigenkänningssystem används för identifieringsändamål. Relevansen av ML inom bildbehandling understryks av dess potential att avsevärt förbättra effektiviteten, noggrannheten och beslutsprocesserna inom dessa kritiska områden.

Bland de många datauppsättningar som används för att jämföra och utvärdera prestandan hos bildbehandlingsalgoritmer sticker den modifierade datamängden från National Institute of Standards and Technology (MNIST) ut. MNIST-datasetet består av 70 000 handskrivna siffror och fungerar som en grundläggande resurs för forskare som vill utveckla och förfina ML-modeller som kan klassificera uppgifter. Datauppsättningens enkelhet men ändå utmanande karaktär gör den till en utmärkt kandidat för att utforska funktionerna i olika ML-metoder, från traditionella algoritmer till avancerade neurala nätverk.

Detta arbete är grundat i det bredare sammanhanget av att utnyttja ML för bildklassificering, särskilt med hjälp av MNIST-datauppsättningen för att undersöka effektiviteten hos olika ML-modeller. Den intrikata uppgiften att klassificera siffror ger inte bara insikter om styrkorna och begränsningarna hos varje modell, utan bidrar också till den pågående diskussionen om hur man bäst implementerar ML-tekniker för bildigenkänningsändamål.

Syfte och frågeställning

Syftet med denna rapport är att utvärdera prestandan hos olika övervakade klassificeringsmaskininlärningsmodeller i samband med bildbehandling, med hjälp av MNIST-datasetet av handskrivna siffror som en fallstudie. Denna undersökning syftar till att identifiera de mest effektiva ML-modellerna för bildklassificeringsuppgifter och köra den bästa i en Streamlit-applikation, och därigenom lära sig viktiga aspekter om optimering av bildbehandlingsapplikationer i teknikdrivna sektorer.

För att uppfylla syftet kommer följande två frågeställningar att besvaras:

1. Kan vi uppnå 90 % noggrannhet i en modellbaserad app när vi klassificerar handskrivna siffror från MNIST-datauppsättningen?
2. Hur påverkar beräkningskomplexiteten hos dessa modeller deras praktiska användbarhet i bildbehandlingsuppgifter?

1.1 Begränsnigar

För att hitta den bästa maskininlärningsmodellen för att korrekt gissa vad nya handskrivna bilder är, med sikte på en framgångsfrekvens (F1-score) över 90 %, togs beslutet att koncentrera sig på tre specifika modeller: Random Forest, KNN och Support Vector Machines (SVM). Detta gjordes för att fokusera på modeller som är bra på att göra korrekta förutsägelser och kan arbeta tillräckligt snabbt för att användas i en Streamlit-app.

1.2 Översikt

Denna uppgift börjar med en introduktion och avgränsning av problemet. Den går sedan in mer i detalj på de relevanta allmänna och specifika teorier som är nödvändiga för att förstå uppgiftens omfattning. Därefter beskrivs de experiment och metoder som används. Därefter presenteras och tolkas resultaten. Sammanfattningsvis sammanfattar uppgiften resultaten och föreslår vägar för framtida utforskning och förfining av modellen.

2 Teori

2.1 Klassificierarmodeller

I detta avsnitt presenteras teori som är relevant för att förstå projektets sammanhang.

Vi tittade på en rad olika klassificerarmodeller för att hantera en övervakad klassificeringsuppgift med flera klasser och här utforskar vi styrkor och svagheter hos både de modeller vi använde och andra som vi skulle kunna använda i framtiden.

2.1.1 Random Forest Classifier

2.1.1.1 Överview och tillämpning i Image Recognition

Random Forest (RF) Classifiers fungerar genom att konstruera en mängd beslutsträd vid träningstillfället och mata ut den klass som är läget för klasserna för de enskilda träden. I samband med bildigenkänning är de skickliga på att hantera högdimensionella data och kan urskilja komplexa mönster genom att ta hänsyn till olika egenskaper hos bilderna, vilket gör dem kraftfulla för att klassificera bilder i fördefinierade kategorier. (Breiman, 2001, s. 2).

2.1.1.2 Styrkor och begränsingar

Enligt Breiman (2001) är en av de främsta styrkorna hos RF Classifiers i bildigenkänning deras robusthet mot överanpassning, särskilt när det handlar om många beslutsträd. Deras komplexitet kan dock leda till betydande beräkningskrav, särskilt med mycket stora datamängder som är vanliga i bildbehandlingsuppgifter. (Breiman, 2001, s. 2).

Som vi kan se i fuguren här tog vår RF-tunede modell initialt över 8 timmar att köra med för många anpassningar med tanke på våra initiala hyperparameterinställningar.

Table 1: GridSearch CV på Random Forest tuned - Beräkningsfördröjningar

```
print(f"Validation Accuracy of the best model: {val_accuracy}")
print(f"Validation F1 Score of the best model: {val_f1}")

print("Best parameters for Random Forest:", rf_grid_search.best_params_)
print("Best score for Random Forest:", rf_grid_search.best_score_)
```

504m 16.1s

```
Fitting 3 folds for each of 18 candidates, totalling 54 fits
```

2.1.2 Linear Support Vector Classifier

2.1.2.1 Överview och tillämpning i Image Recognition

Support Vector Machines (SVM), och specifikt Linear Support Vector Classification (Linear SVC), representerar en kraftfull delmängd av övervakade inlärningsalgoritmer som används för klassificerings- och regressionsuppgifter. Linjär SVC är utformad för att hitta det hyperplan som bäst separerar olika klasser i funktionsutrymmet genom att maximera marginalen mellan klasspunkter och beslutsgränsen. (Cortes, C., Vapnik, 1995). Denna linjära klassificerare är särskilt effektiv när det gäller högdimensionella data, vilket gör den till en stark kandidat för vår uppgift, där det är avgörande att hantera komplexa relationer inom data utan överanpassning.

2.1.2.2 Styrkor och begränsningar

En av de främsta styrkorna med linjär SVC är dess robusthet i högdimensionella utrymmen, även när antalet dimensioner överstiger antalet prover. Detta beror delvis på att optimeringsproblemet för linjär SVC endast beror på datapunkterna nära beslutsgränsen (support vectors), vilket gör det mindre

benäget att överanpassas. Dessutom är linjär SVC mångsidig, med dess effektivitet som inte begränsas av linjär separerbarhet av data, tack vare implementeringar av kärntrick i andra typer av SVM:er. Linjär SVC har dock begränsningar, inklusive känslighet för skalan för indata, vilket kräver noggranna förbearbetningssteg som normalisering. Det handlar också om att välja en lämplig regulariseringsparameter och förlustfunktion, vilket avsevärt kan påverka modellens prestanda och beräkningseffektivitet. (Raj, 2022)

2.1.3 K-Nearest Neighbors

2.1.3.1 Överview och tillämpning i Image Recognition

K-Nearest Neighbors (KNN) är en enkel algoritm som används för både klassificerings- och regressionsuppgifter, men den används främst i klassificeringsscenarier, inklusive bildigenkänning. Den fungerar genom att identifiera de k närmaste träningsexemplen till en given testpunkt och förutsäga resultatet baserat på majoritetsomröstningen bland dessa grannar. Denna enkelhet gör att KNN kan användas effektivt i olika bildigenkänningssammanhang, från handskriftsdetektering till ansiktsigenkänning, hantera problem med flera klasser med lätthet och anpassa sig till olika datauppsättningar utan behov av omfattande modellträning eller justering. (Grover J & Rishabh M., 2021, s. 138-139).

2.1.3.2 Styrkor och begränsningar

Men enligt Grover (2021, s. 139), även om KNN är lätt att använda, är det en "lat algoritm" och står inför utmaningar med skalbarhet och effektivitet, särskilt när datavolymen och dimensionaliteten ökar. Algoritmens prestanda kan försämras avsevärt i högdimensionella utrymmen – vilket är vanligt vid bildigenkänning – på grund av dimensionalitetens förbannelse, där avstånden mellan punkterna blir mindre meningsfulla. Dessutom är KNN beräkningsintensivt under prediktionsfasen, eftersom det kräver beräkning av avstånd mellan testpunkten och alla träningspunkter, och är känsligt för irrelevanta funktioner, vilket kräver noggrant val eller minskning av funktioner.

2.2 Utvärderingsmått med fokus på F1-score

Enligt Smolic (2024) används ofta flera nyckeltal för att mäta effektiviteten hos klassificeringsmodeller för maskininlärning. Baserat på hans bedömningar av balansen är dock F1-scoreen det mest relevanta fokuset för denna bildigenkänningsuppgift.

> F1-scoreen tar hänsyn till den här kompromissen och ger ett enda mått för att utvärdera modellens övergripande prestanda. Det är det harmoniska medelvärdet av precision och träffsäkerhet, vilket ger lika stor vikt åt båda takterna. Genom att kombinera precision och träffsäkerhet ger F1-scoreen en balanserad bedömning av modellens förmåga att korrekt klassificera positiva instanser samtidigt som både falska positiva och falska negativa identifieringar minimeras.

2.3 Skalning

Funktionsskalning är ett viktigt förbearbetningssteg i bildigenkänning, särskilt med hjälp av modeller som beskrivs ovan som KNN och linjär SVC. Skalning säkerställer att alla funktioner bidrar lika mycket till modellens prestanda, vilket förbättrar beräkningseffektiviteten och potentiellt förbättrar modellens prestanda genom att eliminera brus. Det är viktigt att skapa en datauppsättning som fångar ett brett spektrum av variationer, både inneboende i objekten av intresse (inneboende variabilitet) och de som härrör från deras miljöer (extern variabilitet), såsom förändringar i belysning, storlek. Denna variabilitet utgör en betydande utmaning, eftersom objektet vi försöker identifiera eller klassificera inte ser likadant ut i alla bilder (Hulström, 2013). Detta exempel är hämtat från s. 7 i: Hultström, K. (2013). *Bildbaserad hjuldetektering med hjälp av slumpmässig skogsklassificering*. Lund: Lunds Tekniska Högskola.

3 Metod

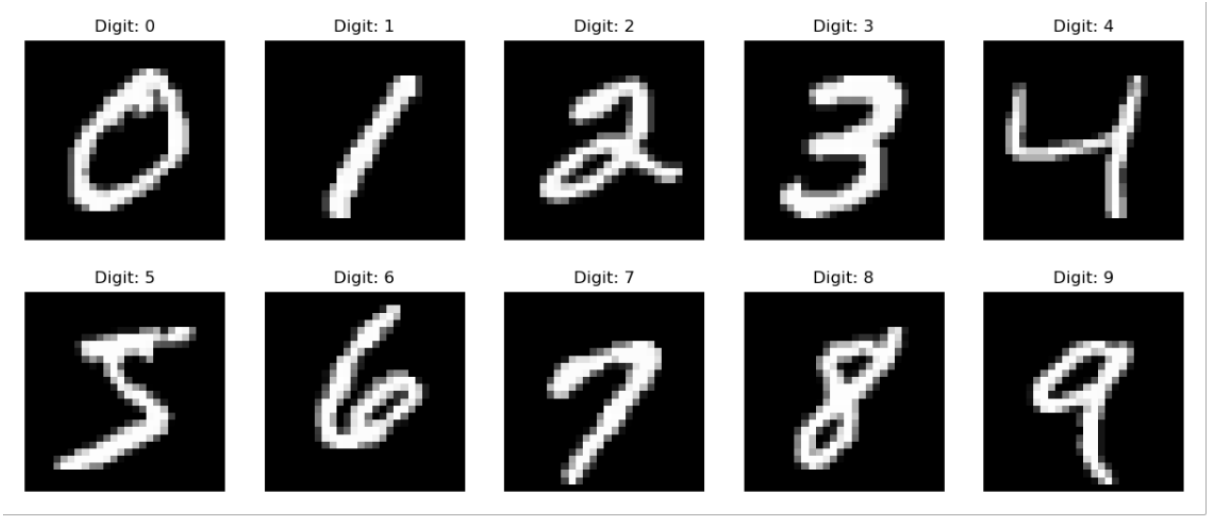
I det här kapitlet beskrivs vår steg-för-steg-metod för projektet, som omfattar dataursprung, dataförberedelse med Exploratory Data Analysis (EDA), modellval, finjustering av parametrar och en grundlig presentation av den slutliga klassificeringsmodellen.

3.1 Materialets ursprung och förberedelse

Datamängden i fråga kommer från Modified National Institute of Standards and Technology (MNIST), som är känt för sin användbarhet vid benchmarking av bildklassificeringsalgoritmer inom maskininlärningsgemenskapen. Den här datauppsättningen består av 70 000 gråskalebilder av handskrivna siffror, var och en noggrant märkt för att underlätta övervakad inlärning. Förberedelsefasen omfattade normalisering av pixelvärden till en skala [0, 1] för att optimera effektiviteten i modellträningen. Till en början körde vi PCA för att hjälpa till med beräkningshastigheten, men bestämde oss sedan istället för att minska provstorlekarna. När det gäller MNIST representerar var och en av de 784 funktionerna ett pixelvärde, och även om dessa funktioner var för sig kanske inte är särskilt informativa, representerar de tillsammans meningsfulla mönster. Så logiken var att med tanke på att datauppsättningen är relativt balanserad när det gäller antalet funktioner jämfört med antalet exempel, vilket hjälper till att tillämpa maskininlärningsalgoritmer direkt utan det omedelbara behovet av dimensionsminskning.

Noterbart är att datauppsättningens höga kvalitet och fullständiga avsaknad av saknade värden – vilket avviker från normen i verkliga datauppsättningar – pekade på en anomali i datainsamling och hantering där denna process normalt tar mycket längre tid.

Table 2 Exempel på siffror




```
Array Shape: (70000, 784)
Array Shape: (70000,)
Data Type: float64
Memory Usage: 428750.0 KB
```

Table 3: Dataprover, form, typ och minne

Ovan kan vi se några av provsiffrorna och variationen i form och storlekar. MNIST-datauppsättningen har 784 funktioner per bild, vilket motsvarar pixlarna i 28x28 gråskalebilder, där pixelintensiteten

sträcker sig från 0 till 255 och ofta normaliseras till flyttal mellan 0 och 1 för maskininlärning. Den här normaliseringen, som är nödvändig för kompatibilitet med inlärningsalgoritmer och förbättrad beräkningseffektivitet, flyttar datatypen till flyttal, vilket resulterar i att datauppsättningen upptar drygt 0,4 GB minne.

Datauppsättningen delades upp i tre distinkta uppsättningar: en träningsuppsättning som bestod av 70 % av data, en valideringsuppsättning som omfattar 15 % och en testuppsättning som också står för 15 %. Stratifieringen säkerställer att alla uppsättningar upprätthåller en representativ andel sifferprover, vilket bidrar till en robust utvärdering av modellen.

3.2 Insikter från EDA och använda verktyg

Varje bild i MNIST-datauppsättningen är 28x28 pixlar, dessa har plattats ut till 784-dimensionella vektorer före skalning. Skalning är avgörande innan för att säkerställa att varje funktion bidrar lika mycket till avståndsmåttet.

Exploratory Data Analysis (EDA) började med att använda NumPy för datavrängning och Matplotlib för visuell utforskning. EDA avslöjade en viss grad av klassobalans och distinkta mönster för specifika siffror som "1", insikter som en PCA-visualisering ytterligare validerade genom att avslöja kluster som gynnade enklare former.

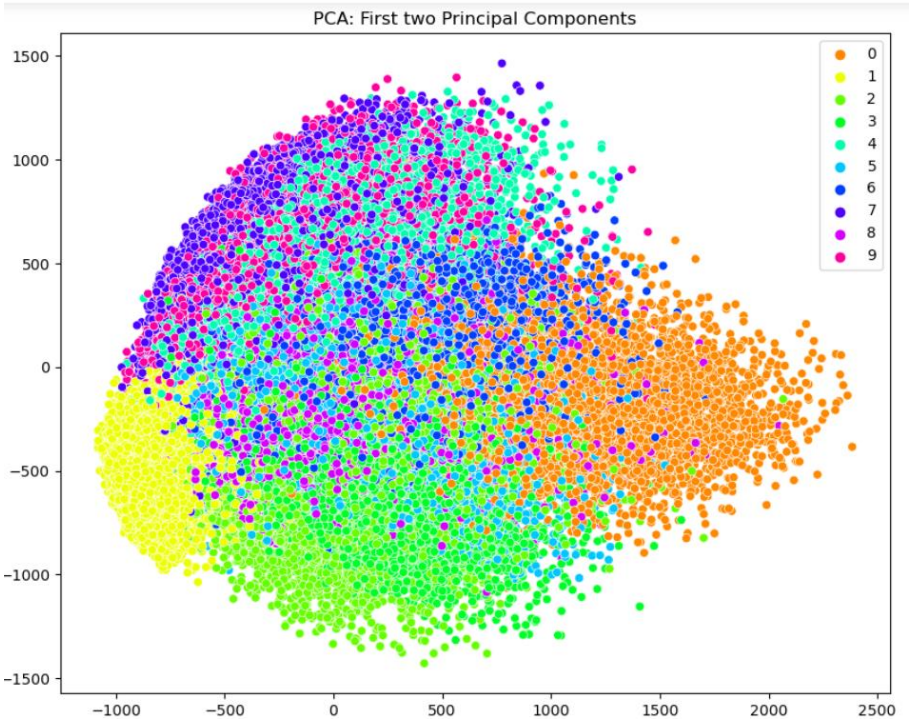


Table 4: PCA Kluster

Med tanke på att MNIST-datauppsättningen är relativt ren och standardiserad utan saknade värden, handlar det vi ser här mer om att använda en reducerad dimension för att visa oss hur data är strukturerade och var datauppsättningen verkar bilda renare - distinkta men inte separata - grupperingar runt vissa bilder som 0 och 1, vilket kan indikera en potentiell predisposition för att dessa klasser lättare ska kunna identifieras när vi bygger vår modell.

3.3 Modellval och åtgärdsbegränsningar

Vår modellurvalsprocess övervägde initialt RF Classifiers och Support Vector Machines, i form av en linjär SVC, på grund av deras beprövade robusthet och meritlista i klassificeringsutmaningar. Båda klassificeringsmodellerna är ganska komplexa och särskilt bra på att hantera datauppsättningar med hög dimensionalitet som MNIST, samtidigt som de kan generalisera väl för att undvika överanpassning. Detta är avgörande för bilddatauppsättningar, där variationer i pixelvärden kan introducera brus. Med det sagt resulterade den omfattande urvalsstorleken och komplexiteten i MNIST-

datauppsättningen i beräkningssvårigheter, särskilt med RF- och SVC-modellerna som brottades med hög dimensionalitet och långa beräkningstider. Dessa utmaningar ledde till att man utforskade K Nearest Neighbors som ett beräkningseffektivt alternativ. När man minskade urvalsstorleken och finjusterade hyperparametrarna för KNN- och RF-modellerna blev det uppenbart att Random Forest framstod som den överlägsna modellen baserat på F1-scorenoggrannhet och bearbetningshastighet, och överträffade de andra två modellerna.

3.4 Utvärdering av modellens prestanda

Utvärderingsramverket för modellprestanda var omfattande och utnyttjade noggrannhet, precision, träffsäkerhet och särskilt F1-scoreen för att mäta effektiviteten hos varje modell på ett heltäckande sätt. F1-scoreen prioriterades på grund av dess balanserade mått på precision och träffsäkerhet, vilket var vettigt med tanke på sammanhanget och för att ta itu med datauppsättningens problem med obalans i klassen. Verktyg som tillhandahölls av Scikit-learn-biblioteket underlättade djupgående metriska beräkningar, vilket möjliggjorde en detaljerad undersökning av modellens prestanda i olika konfigurationer. Vi såg till exempel att vi genom hyperjustering ökade antalet beslutsträd från 100 till 200 och djupet på tre från 10 till 20, vilket förbättrade modellens prestanda med 1 % (Géron, A., 2019, p.199).

3.5 Val av slutlig ML-modell och härledda insikter

Minskningen av urvalsstorleken som balanserades med hyperparameteroptimering av KNN- och Random Forest-modellerna markerade Random Forests oöverträffade balans mellan beräkningseffektivitet och modellnoggrannhet, med en f1-score på 95 %, vilket gör den till den bästa modellen. Denna process belyste EDA:s avgörande roll när det gäller att avslöja grundläggande dataattribut, såsom obalans i klass, och underströk skalning för att effektivisera modellträning och tolkning. Detta arbete belyste inte bara de faktorer som påverkar modellval och dataförberedelse utan visade också syftet med dimensionalitetsreduktionstekniker för att hantera de inneboende utmaningarna med bildklassificering. En viss överanpassning av RF-modellens träningsdata kvarstod med trimning (99 %), den lyckades fortfarande prestera bra på både validerings- och testuppsättningar, med 96 % respektive 95 % på f1-score. Så vi valde den här tunede RF-modellen som den bästa, men den verkar orsaka problem senare med distributionen på Streamlit-applikationen.

3.6 Streamlit APP

Därefter flyttade vi till distribution på vår lokala server och skapade en python-fil (bättre för att köra Streamlit) och laddade in den Random Forest-tunede modellen i detta tillsammans med de nödvändiga Scikit Image- och Numpy-biblioteken. Vi körde en uppsättning förbearbetningssteg för att förbereda oss för nya dataexempelbilder som innebar att ställa in en rad parametrar som skulle justera storlek, form, ljusstyrka, position och förgrund/bakgrund för dessa bilder för att efterlikna de data som vår modell tränade på med hjälp av Mnist-datauppsättningen. Mer specifikt körde vi kod från scikit-bildbiblioteket för att begränsa appen så att den bara skulle behöva hantera data av mnist-typ, med andra ord gråskala, svart siffra på vita bakgrundsbilder, ändrad till 28X28 med 1 dimension som har intensitetsnivåer från 0 till 255. Detta är vad vår modell tränades på och därför borde den kunna fungera om vår app korrekt kan förbehandla alla färgbilder (med varierande grad av opacitet) för att bli gråskala med en 28X28 1-dimensionell form. Slutligen, innan vi kör appen, ställer vi in den på att skala nya inkommande avbildningar med StandardScaler som vi gjorde med modellen.

4 Resultat

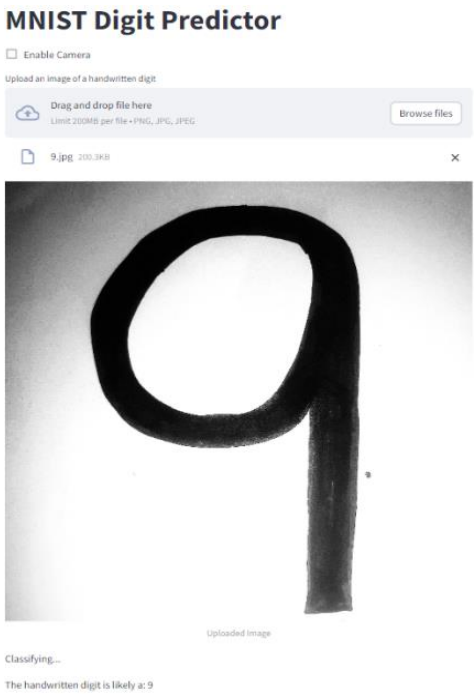
I denna avsnitt presenteras en jämförelse av F1-score för olika modeller, inklusive linjär SVC, tuned KNN och RF (Random Forest), över tre olika dataset: träningssetet, valideringssetet och testsetet. Genom att analysera dessa resultat kan vi bedöma modellernas prestanda och deras förmåga att generalisera från träning till validering och slutligen till okända data i testsetet. Denna analys ger insikter i hur väl varje modell hanterar överanpassning och dess potential för praktisk tillämpning.

F1-score för modeller			
	Train Set	Validation Set	Test Set
SVC linjär	95%	87%	
KNN-tuned	100%	93%	
RF-tuned	99%	96%	95%

Table 5: F1 Scores för de 3 modellerna

- Först valde vi den RF-tunede modellen som den modell som hade bäst f1-score på 95 % på de osedda data från testuppsättningen. F1-testresultaten för RF-modellen indikerar en balanserad prestanda mellan precision och träffsäkerhet, vilket är bra i scenarier där både falska positiva och falska negativa är lika oönskade. Klasser/siffror 0, 1 och 6 visade särskilt starka prestanda för alla mått, vilket indikerar att modellen är mycket exakt och balanserad när det gäller att förutsäga dessa klasser. Även om den övergripande prestandan är stark, har klasserna 7, 8 och 9 något lägre mätvärden jämfört med andra. Klass 8 har den lägsta återkallelsen, vilket tyder på att den har fler falska negativa resultat, och klasserna 7 och 9 har något lägre precision, vilket indikerar en högre falsk positiv frekvens för dessa förutsägelser.

STREAMLIT APP



- Distributionen av appen med hjälp av denna förbehandlade RF-baserade modell fungerade dock inte bra, utan kände bara igen en siffra från tio. Så vi provade det med KNN-modellen, och den fungerade inte heller bra.

- Så till slut bestämde vi oss för att problemet kunde vara det faktum att både KNN- och RF-modellerna överanpassade träningsdata och därmed inte var tillräckligt robusta för att hantera nya siffror utan att betydligt mer tid spenderades på att förbehandla bilden, vilket vi inte hade tid att göra. Med tanke på att den linjära SVC-modellen är särskilt skicklig på att inte överanpassa och det faktum att den inte överanpassade, bestämde vi oss för att prova den istället. Till en början kände den bara igen 2 siffror, men efter att ha lagt till ett tröskelvärde på 0,5 (vilket hjälpte till att definiera beslutsgränserna för klassen för SVC Linear-algoritmen) förutspådde den korrekt 9 av 10 siffror. Tröskelvärdet hjälpte till att klargöra vilka pixlar som skulle betraktas som en del av en siffra jämfört med bakgrunden, vilket förbättrade modellens förmåga att skilja mellan klasser.

5 Slutsatser-Diskussion

5.1 Fråga 1: Kan vi uppnå 90 % noggrannhet på en modell?

Vi lyckades generera en lokal server Streamlit-applikation med 90% noggrannhet, med 9 av 10 klasser eller siffror korrekt identifierade. Med det sagt kunde vi säkert ha fortsatt om vi hade haft tid att hyperjustera den linjära SVC-modellen, testa på hela datauppsättningen, testa med neurala nätverksmodeller som CNN och dessutom testa med fler förbehandlingssteg i python-skriptet som Otsu-tröskeln.

5.2 Fråga 2: Hur fungerar modeller i verkligheten?

Utforskningen av andra modeller, som KNN och så småningom den linjära SVC, och justeringen av förbehandlingssteg, som att införa ett tröskelvärde för att hjälpa till att definiera klassbeslutsgränser bättre, var steg som togs för att ta itu med dessa utmaningar. Dessa ansträngningar ledde till förbättrad verklig applikationsprestanda, vilket belyser vikten av kontinuerlig modellutvärdering och anpassning till specifika applikationssammanhang. Framgångsrika bildigenkänningsprojekt, som de som använder MNIST-datauppsättningen, är beroende av synergin mellan maskininlärningsmodeller och utvecklingsramverken, till exempel Streamlit för webbapplikationer. Det är viktigt att algoritmens inställningar och funktioner passar bra ihop med bildigenkänningsverktygen och Streamlit för att säkerställa smidig integration och topprestanda.

5.3 Lärdomar och nästa steg:

Förbehandlingens betydelse: Vår erfarenhet understryker den kritiska roll som bildförbehandling spelar för att förbereda data för modellindata, särskilt när det gäller variationer i färg, storlek och orientering. Det är viktigt att matcha förbearbetningsstegen med modellens träningsdata.

Modellval och överanpassning: Utmaningen med RF- och KNN-modellerna belyste frågan om överanpassning i maskininlärningsdistributioner. Om du väljer en modell som är mindre benägen att överanpassa, till exempel linjär SVC i ditt fall, kan generaliseringen till nya data förbättras.

Parameterjustering: Framgången med tröskeljusteringen illustrerar hur finjustering av modellparametrar och förbearbetningssteg dramatiskt kan förbättra prestandan.

Med tanke på förbättringarna med den linjära SVC-modellen kan ytterligare förbearbetning av förbehandlingspipelinen och överväga ytterligare parameterjusteringar eller till och med ensemblemetoder ge ännu bättre resultat. Kontinuerlig testning med en mängd olika avbildningar och inkrementella justeringar kommer att vara nyckeln till att optimera prestanda för verkliga program.

6 Teoretiska frågor

1. Kalle delar in sin data i "Träning", "Validering" och "Test", vad används varje del till?
   A. Träning används för att passa ett visst antal modeller med hjälp av färdiga inställningar (standardhyperparametrar) och sedan utvärderar vi dessa modeller med hjälp av ett relevant bedömningsmått i valideringsuppsättningen.
   B. Den här validerings- eller undantagsuppsättningen med osedda data används som ett preliminärt test av data för att hjälpa oss att välja den slutliga bästa modellen för distribution. Baserat på poängmåttet kan vi justera hyperparametrarna i enlighet med detta.
   C. Sedan testar vi denna bästa modell med hjälp av osedda testdata för att ge oss en opartisk utvärdering av hur den bästa modellen generaliserar till nya data.

2. Julia delar upp sina data i träning och testning. På träningsdata tränar hon tre modeller; "Linjär regression", "Lassoregression" och en "Random Forest". Hur ska hon välja vilken av de tre modellerna som ska fortsätta använda när hon inte har skapat en explicit "valideringsdatauppsättning"?

   Om Julia inte har skapat en explicit valideringsuppsättning kan hon i stället använda korsvalidering, som delar upp träningsuppsättningen i olika vikningar och tränar/utvärderar fram och tillbaka så att vi kan få ett genomsnittligt prestandamått för varje modell och sedan välja den bästa för testning.

3. Vad är "regressionsproblem? Kan du ge några exempel på modeller som används och potentiella tillämpningar?

   Ett regressionsproblem är ett övervakat inlärningsproblem som förutsäger en kontinuerlig variabel baserat på en rad funktionsvariabler. Linjär regression, åsregression och slumpmässig skogsregression är exempel på regressionsmodeller och dessa kan användas inom fastighetsbranschen för att förutsäga bostadspriser samt finans för att förutsäga börskurser.

4. Hur kan du tolka RMSE och vad används det till: RMSE är Root Mean Squared Error och är ett utvärderingsmått som används av regressionsmodeller för att mäta modellens prestanda genom fel eller skillnad mellan modellens sanna och förutsagda värden.

5. Vad är "klassificeringsproblem? Kan du ge några exempel på modeller som används och potentiella tillämpningar? Vad är en "Förvirringsmatris"? Ett klassificeringsproblem är ett övervakat inlärningsproblem som förutsäger en diskret variabel baserat på en känd etikett och vanligtvis omfattar utdata från en binär klass, men som också kan användas för att leverera sannolikhetsutdata, samt utdata med flera klasser. Klassificeringsmodeller, till exempel modeller som används för klassificering, inkluderar Logistic Regression, Decision Trees och Neural Networks, och används inom områden som skräppostdetektering och sjukdomsverifiering. En förvirringsmatris är en tabell som mäter prestanda för en klassificeringsmodell, genom en matris som jämför korrekta och felaktiga förutsägelser med sanna värden.

6. Vad är K-means-modellen? Ge ett exempel på vad det kan tillämpas på. K-means är en oövervakad klustringsalgoritm som organiserar omärkta data i grupper genom att skapa centroider och mäta avstånd mellan datapunkterna och dessa centroider. Detta kan användas för att organisera aktieportföljer och till och med för kundsegmentering i marknadsföringskampanjer.

7. Förklara (gärna med ett exempel): Ordningstalskodning, one-hot-kodning, dummyvariabelkodning. Se mappen "l8" på GitHub om du behöver uppdatering.

A. Ordningstalskodning tilldelar tal till kategorier med en naturlig ordning (t.ex. "låg" = 1, "medel" = 2, "hög" = 3), vilket är användbart för rangordnade data som husförhållanden som påverkar priserna.
B. One-Hot-kodning omvandlar kategorier till binära kolumner (1 för närvaro, 0 för frånvaro). För "färg" med rött, grönt, blått får du separata kolumner för varje, lämpliga för osorterade kategorier.
C. Dummy Variable Encoding är som one-hot-kodning för N-kategorier som skapar N-1-kolumner för att undvika funktionsöverlappning, och är mycket bra i regressionsmodeller för att införliva kategoriska data effektivt.

8. Göran hävdar att datan är antingen "ordinal" eller "nominell". Julia säger att detta måste tolkas. Hon ger ett exempel på att färger som {röd, grön, blå} i allmänhet inte har någon nominell ordning, men om du har en röd skjorta är du vackrast på festen (ordinal) – vem har rätt?

Julia har rätt. Om data betraktas som ordningstal eller nominella beror på dess kontext och hur de tolkas. Även om färger som {röd, grön, blå} i allmänhet är nominella utan någon inneboende ordning, kan de i specifika sammanhang (som Julias exempel på attraktivitet på en fest) få en ordningskaraktär där ordningen är viktig. Kontext och tolkning är nyckeln.

9. Kolla in följande video om Streamlit: https://www.youtube.com/watch?v=ggDa-RzPP7A&list=PLgzaMbMPEHEx9Als3F3sKKXexWnyEKH45&index=12 Och svara på följande fråga: - Vad är Streamlit och vad kan det användas till?

Streamlit är ett Python-bibliotek som gör det enkelt och gratis att skapa och dela appar för maskininlärning. Det fungerar som ett distributionstillägg med låg kod till Python.

7 Självutvärdering

1. Vilka utmaningar du har haft under arbetets gång och hur du har hanterat dem. Att försöka bearbeta många nya koncept utan att bli förvirrad, och att försöka göra de enkla sakerna bra. Att hantera kaninhålet av fel och acceptera begränsningar som beräkningstid och tidsfrister.

2. Vilket betyg du tycker att du ska ha och varför.
   Nära VG då jag byggde solida modeller och hittade kreativa lösningar för att få appen att fungera. Jag lärde mig också väldigt mycket genom att felsöka fel själv och med mina klasskamrater.

3. Något du vill lyfta fram för Antonio?
   Jag tycker att det var en jättebra övning. Det kan vara intressant att titta på olika sätt att sammanfoga rapporten och koden i ett dokument, samt en YTube-video om hur du använder Visual Code och hur du söker efter fel mellan olika plattformar/bibliotek som mellan Jupyter Notebook och Scikit Learn med Python och Scikit Image.

Appendix A

Table 6: Random Forest Tuned Klassificering Rapport

RandomForestClassifier

	precision	recall	f1-score	support
0	0.97	0.99	0.98	305
1	0.97	0.98	0.98	352
2	0.96	0.92	0.94	291
3	0.96	0.92	0.94	289
4	0.96	0.93	0.95	288
5	0.95	0.97	0.96	267
6	0.95	0.99	0.97	278
7	0.95	0.96	0.95	301
8	0.96	0.96	0.96	313
9	0.91	0.91	0.91	316
accuracy			0.95	3000
macro avg	0.95	0.95	0.95	3000
weighted avg	0.95	0.95	0.95	3000

Källförteckning

1. Breiman, L. (2001*).* Random forests. *Machine Learning* 45, 5–32. University of California, Berkeley: Department of Statistics, UC Berkley.

2. Cortes, C., Vapnik, V. (1995). Support Vector network. *Machine learning* 20, 273–297.

3. Géron, A. (2019). Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow (2nd edition). O'Reilly Media, Inc.

4. Grover, J & Rishabh, M. (2021). Sculpting data for ML. The first act of machine learning. (First edition). Grover J & Rishabh M.

5. Hultström, K. (2013). *Image-based wheel detection using Random Forest classification*. Lund: Lunds Tekniska Högskola.

6. Raj, A., 2022. "Everything about SVM Classification – Above and Beyond". Hämtad 18 mars 2024 från https://towardsdatascience.com/everything-about-svm-classification-above-and-beyond-cc665bfd993e.

7. Smolic, Hrvoje. (2024). "Understanding *the importance of F1 score in machine learning*". Hämtad 16 mars 2024 från https://graphite-note.com/understanding-the-importance-of-f1-score-in-machine-learning.