A machine learning project for handwritten digit classification

Joakim Salomonsson

EC Utbildning

Maskininlärning

2025-03-21

Abstract

A machine learning exercise where I train and evaluate three different classification models; RandomForest, DecisionTree, and LogisticRegression. I train the models using various methodologies to find the best-performing model on the data set MNIST. Once trained and evaluated I apply this model in a streamlit app where a user can input handwritten digits through multiple options to predict the digit.

Innehållsförteckning

1. Inledning

Syftet med denna rapport är att presentera mitt arbete och dess resultat inom kursen Maskininlärning med datasettet MNIST. Fokus ligger på utvärdering för modellval kring klassificeringsproblem där jag har valt 3 olika modeller att utvärdera. Utöver modellval har jag även valt att utvärdera dessa modeller genom att dela upp datan i tränings och valideringsdel samt att inte dela upp datan i en valideringsdel, mer om detta under avsnittet metod. Jag kommer att besvara följande frågeställning:

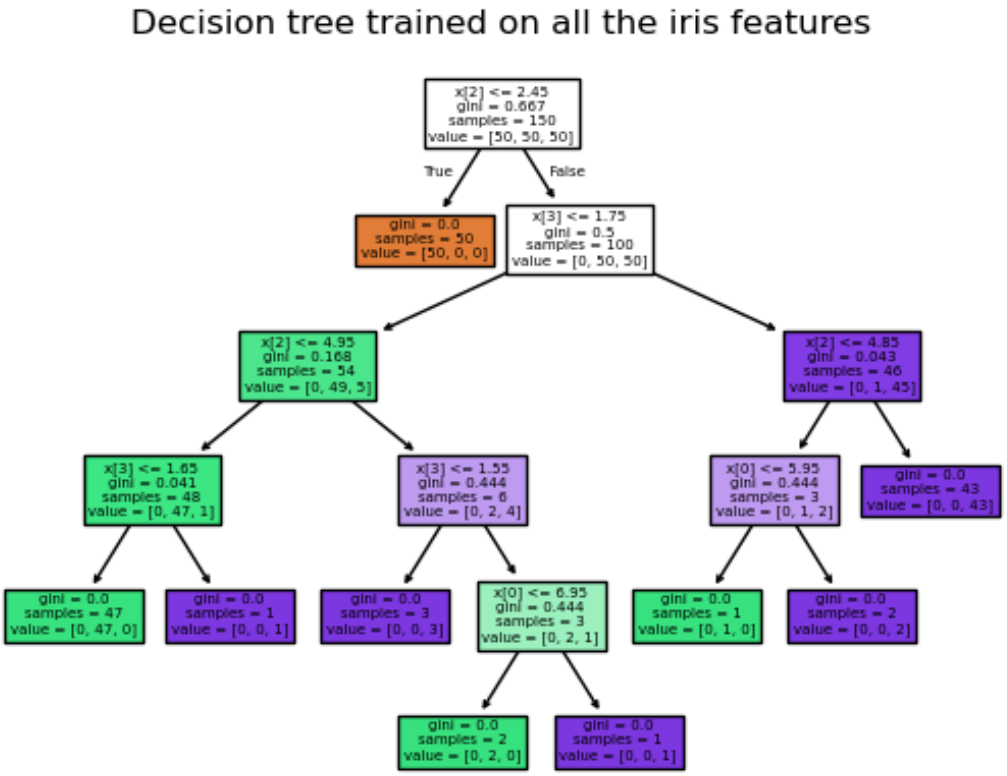
Vilken modell har bäst accuracy på valideringsdatan utifrån vår metod?

2. Teori

2.1. Klassifikations-modeller

2.1.1. DecisionTreeClassification

DecisionTreeClassificaction är en av de modeller jag har valt att använda i mitt maskininlärningsflöde för MNIST datasettet. DecisionTree modellens mål är att lära sig regler om hur modellen ska fatta beslut och prediktera klassifikationssproblem (i mitt fall, fungerar även med regression) och dessa beslut grundas och defineras av datasettets "Features". Se bild nedan om hur ett beslutsträd kan se ut, jag använder mig av det kända datasetet "Iris", bild tagen från scikit learn.



https://scikit-learn.org/stable/auto_examples/tree/plot_iris_dtc.html

(Scikit-learn, 2025)

2.1.2. RandomForestClassification

RandomForestClassification är ytterliggare en av de modeller jag har valt att använda i vårt maskininlärningsflöde. RandomForestClassification är en så kallad "Ensemble Method". "Ensemble

Method” definieras av att den använder sig av flera DecisionsTrees på olika delar av datan för att sedan kika på genomsnitten av dessa. Enligt SciKit learn *(Scikit-learn, 2025)* så har DecisionTree allmänt ganska hög varians och tendens till overfitting, dock när man kombinerar flera DecisionsTrees till en RandomForest brukar den få en minskad varians och därmed brukar ofta vara en bättre modell.

2.1.3. LogisticRegression

LogisticRegression är den tredje och sista modellen jag valde att utvärdera på datasettet MNIST. LogisticRegression är en klassificationsmodell som predikterar problem genom att binärt utvärdera features och klassifiera dessa.

Spiceworks (2024)

2.2. Accuracy

Accuracy är ett värde som används i samband med klassificationsproblem, den mäter ration i hur många gånger en modell lyckas prediktera korrekt klass.

(Géron, 2019, s.2)

2.3. Confusion Matrix

En Confusion Matrix är även ett verktyg som kan används för att mäta hur väl en modell har presterat för ett klassifikationsproblem. I en Confusion Matrix kan du se hur många gånger modellen har predikterat korrekt men även när den har predikterat inkorrekt.

(Géron, 2019, s.2)

Se bild för exempel:

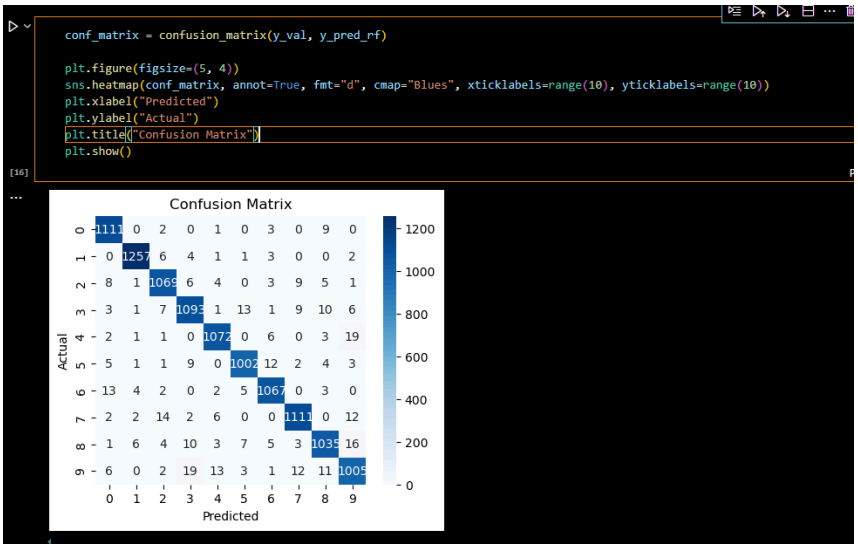


Bild tagen från mitt arbete, specifikt Kunskapskontroll2 train_val_test MASTER.ipynb

2.4. Standard Scaler

En Standard Scaler används för att transformera and standardisera datan. Beroende på vad för modeller som valts kan det vara viktig att behöva standardizera datan.

(Géron, 2019, s.69)

2.5. GridSearch

GridSearch används för att automatisera hyperparameter kalibrering för att enkelt hitta de bästa parametrarna utan att manuellt behöva gå igenom alla.

(Géron, 2019, s.76)

2.6. RandomSearch

Liknande till GridSearch fast RandomSearch, som det låter, gör en slumpmässig sökning på de bästa hyperparameterna för att minimera användet av resursera och att snabba upp processen.

(Scikit-learn, 2025)

2.7. Joblib

Joblib används för att spara modellerna som har tränats för att sedan kunna laddas upp igen utan att behöva träna om modellerna.

(Géron, 2019, s.75)

3. Metod

Jag har valt att utvärdera tre modeller, RandomForest, DecisionTree samt LogisticRegression. Utifrån dessa tre modeller ville jag även undersöka olika tillvägagångssätt när det gäller att hitta hyperparametrar samt att dela upp datan i träning och validering utöver test-set. Jag bestämde tidigt in i min undersökning att ifall det tar längre tid än ca 30 minuter att träna mina modeller skulle jag avbryta. Detta eftersom syftet med uppgiften är att lära sig skapa och bli bekant med maskininlärningsflöden och att sitta att vänta > 30 minuter är oeffektivt.

Jag inledde med att utvärdera en RandomForest modell utan att kalibrera parametrar efter att jag hade delat upp datan i träning, validering och test och för att sedan prediktera på valideringsdata och fick fram en bra accuracy (0.96). Därefter gick jag vidare och utvärderade DecisionTree med samma metodik på valideringsdatan men fick en sämre accuracy (0.86).

I nästa steg ville jag testa att använda Gridsearch på båda dessa modeller. Jag testade både cv 5 och cv 3 men det tog alldelles för lång tid eftersom det är 70000 rader. Jag avbröt körning efter ca 20 minuter.

Jag hittade Randomsearch på sckitlearns dokumentation och testade denna metod istället för att se ifall den kunde gå fortare. Jag ville fortfarande testa att kalibrera hyperparametrar så jag testade att använda Randomsearch istället men jag avbröt även körningen här efter ca 15 minuter.

Inför nästa test bestämde jag att jag skulle ta en mindre del av datan. Nu endast 10000 rader istället för hela datasettets 70000 rader och använda GridSearch på RandomForest. Jag använde mig av endast 10000 rader för att minska tiden det ska ta att köra GridSearch och för att hitta bra hyperparametrar. På sample-datan med GridSearch fick jag en accuracy på 0.9544.

Nästa test körde jag en mindre del av datan med endast 10000 rader fast med RandomSearch denna gång och på RandomForest, det gick på ungefär 30sek och jag fick en accuracy på 0.9543, lite sämre än när jag använde GridSearch men betydligt snabbare fitting.

Jag applicerade samma metodik på DecisionTree, med en small sample size på 10000 rader och testade både RandomSearch och GridSearch och fick följande accuracy:

DecisionTree small sample size accuracy 0.7632 med GridSearch.

DecisionTree small sample size accuracy 0.7631 med RandomSearch.

Jag testade och undersökte även Logistic regression där jag initialt körde en CV 3 och började med 500 iterations, ingen specifik solver men den blev ej “converged”. Jag testade istället med 2000 iterations samt ‘saga’ som solver samt standardiserade datan med en StandardScaler. Efter 15minuter hade den inte körts klar och jag valde att avbryta. Även ifall den inte blev converged med max 500 iterations fick jag en accuracy på 0.9122.

Jag undersökte även Logistic regression med träning, validering samt test data uppdelningsmetodiken. Började med 500 iterations och ingen specific solver och fick resultatet “failed to converge” efter 30 sec. Testade även med 2000 iterations som även blev “failed to converge” efter 2min 20 sec. Testade sedan med en standard scaler och 2000 iterations. Det gick väldigt mycket snabbare med standard scaler och redan efter 12 sekunder var den klar.

Efter alla dessa undersökningar valde jag att gå vidare med small sample RandomForest metodiken där vi använde oss av 10000 rader samt GridSearch. Även ifall detta tillvägagångssätt hade marginellt sämre accuracy än träning, validering samt test metodiken med RandomForest modellen. Jag gick vidare med detta för att jag har kalibrerat hyperparameterna på denna modell, även ifall det undersöktes på en liten del av datan.

Därefter tränade jag om min modell på hela data settet med hyperparametrerna jag fick från sample-settet och detta tog endast två minuter och då fick jag en accuracy på testdatan som var 0.97.

Jag har även använt streamlit för att bygga en sifferklassificierings app. Till att börja med skapade jag en app där användaren kunde antingen ladda upp en bild från datorn eller ta bild med hjälp av den inbyggda webb-kameran i datorn. Tidiga versioner av appen gav jag användaren möjlighet att välja mellan tre olika modeller att prediktera siffror, RandomForest, DecisionTree och LogisticRegression.

Eftersom jag i min tidiga version av streamlit applikationen valde att ha alla tre modeller, så finns det gamla notebooks i min github där jag gått hela vägen och predikterat på test-datan. Men i min slutgiltiga inlämning (i mina MASTER + MAIN Notebooks) har jag endast utgått från accuracy på validering eller träningsdatan (vid GridSearch) för att utvärdera och välja modeller. Samt endast predikterat på test datan med modellen jag valde att gå vidare med, se Main notebook.

Slutgiltiga versionen har jag även lagt till en möjlighet för användaren att rita en siffra med hjälp av musen på datorn. Efter många tester valde jag att ta bort DecisionTree och LogisticRegression som alternativ för att prediktera siffror då jag kände att de presterade sämre.

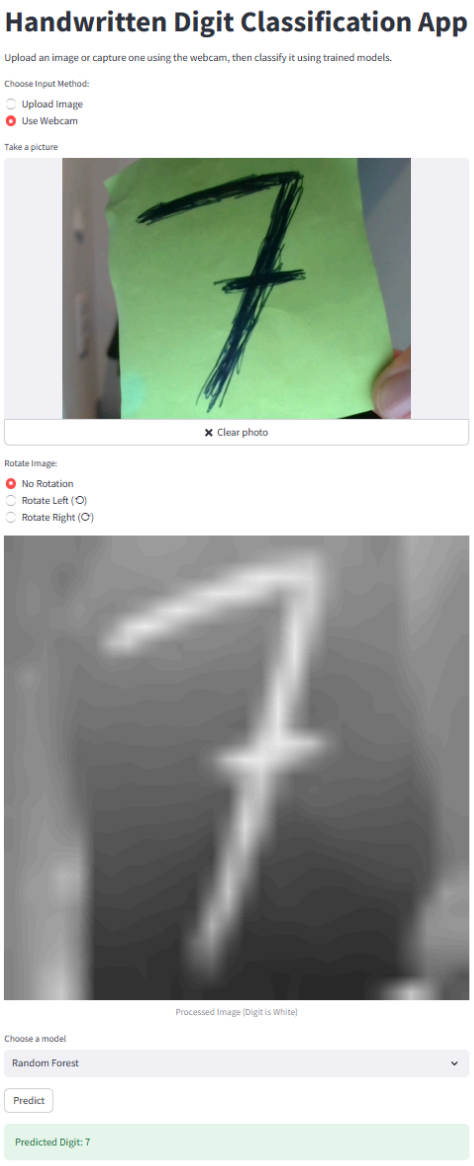
Jag har försökt i flera versioner testa många olika preprocessing tekniker för att förbättra appen och dess förmåga att prediktera siffror utifrån de olika inmatnings-metoderna men kommit fram till att de bästa presterande preprocessingteknikerna för mus-handritade siffror:
Gråskala bilden, ändra till rätt storlek (28, 28), invertera färgerna (vit siffra, svart bakgrund) och ändra till rätt shape.

3. Resultat och Diskussion

Accuracy för olika modeller	
RandomForestSmallSampleCV	0.9544
DecisionTreeSmallSampleCV	0.7632
LogisticRegressionSmallSampleCV	0.9122
RandomForestFullNoCV	0.9663
DecisionTreeFullNoCv	0.8642
LogisticRegressionFullNoCV	0.9155

Tabell 1: Accuracy för de sex valda modellerna.

Tidig version av min Streamlit app där jag lyckades prediktera en korrekt handskriven post-it lapp som tagit med hjälp av webb-kameran.



Senaste versionen av min Streamlit app där jag även lagt till ett alternativ att kunna rita en siffra med dator-musen.

Appen predikterar oftast korrekt på handritade siffror och mer sällan korrekt på webbkamera/uppladdad bild i den slutgiltliga versionen.

Det som kunde förbättras är preprocessingsdelen i min streamlit app. Jag märkte att det eventuellt är olika preprocessings moment som behöver göras beroende på vilken inmatningsmetod användaren vill använda. Om jag normaliserade datan för streamlit appen för RandomForest modellen fungerade handritade siffror med musen bättre men för webcam/uppladdning predikterade den sämre t.ex.

Om jag hade mer tid hade jag kikat på att försöka skapa och skriva specifika preprocessingsfunktioner för varje inmatningsmetod.

Spekulation är att RandomForest modellen fungerar bättre för handritade siffror på en canvas gentemot bilder tagna av en webbkamera samt uppladdade direkt till datorn.

Jag valde att gå vidare med RandomForest modellen där jag hade delat upp datan i en small sample size och kalibrerat hyperparameterna även ifall modellen där jag delade upp datan i träning, validering samt test hade bättre accuracy på valideringssettet. Mitt resonemang var att även ifall det var någorlunda bättre på träning och validering så hade jag inte ändrat några hyperparametrar, och det var en väldigt liten skillnad 0.9663 mot 0.9544.

4. Slutsatser

Vilken modell har bäst accuracy på valideringsdatan utifrån vår metod?

RandomForest med uppdelning av datan i träning, validering samt test utan att använda sig av GridSearch eller RandomSearch. Dock med en minimal skillnad gentemot att använda sig av Gridsearch på ett mindre data set.

Att dela upp datan i mindre delar (10000 istället för 70000) skyndade på processen avsevärt och vi kunde få resultat inom en rimlig tid.

Generellt presterade modellerna som var uppdelade i träning, validering samt test väldigt snarlikt gentemot metoden att bara använda sig av 10000 rader istället för fulla datasettet och köra GridSearch med CV.

RandomSearch och GridSearch accuracy resultat var även väldigt snarlika varandra.

Streamlit slutsats:

Slutsats är att det krävs mer preprocessing av datan för att kunna hantera webbkamera och uppladdad bild än handritad siffra.

5. Teoretiska frågor

1. Kalle delar upp sin data i ”Träning”, ”Validering” och ”Test”, vad används respektive del för?

Kalle tränar sina maskininlärningsmodeller på träningsdatan. Efter att ha tränat modellerna på träningsdatan, så gör Kalle prediktioner på valideringsdatan. Kalle gör därefter en utvärderingen och väljer bästa model. Merge:ar träning + validerings data set och tränar om modellen på båda setten. Slutgiltligen gör ett test på test datan.

2. Julia delar upp sin data i träning och test. På träningsdatan så tränar hon tre modeller; ”Linjär Regression”, ”Lasso regression” och en ”Random Forest modell”. Hur skall hon välja vilken av de tre modellerna hon skall fortsätta använda när hon inte skapat ett explicit ”valideringsdataset”?

Julia behöver använda sig av cross validation när hon tränar sina modeller. Te.x. 5-fold eller 3-fold cross validation. Julia kan därefter mäta sina resultat utifrån RMSE.

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

Ett regressionsproblem predikterar ett värde baserat på summan av input variablerna.

Modeller: Linjär regression, Lasso, Randomforest

Exempel: Churn-analys, kreditrisk.

4. Hur kan du tolka RMSE och vad används det till: $RMSE = \sqrt{1\ n\ \sum(yi - \hat{y}i)\ 2\ n\ 1}$

RMSE är Root Mean Squared Error och det innebär att vi tar medelvärdet av faktiska y-värdet minus det predikterade y-värdet i kvadrat och sen roten ur för att eliminera negativa värden

5. Vad är ”klassificieringsproblem? Kan du ge några exempel på modeller som används och potentiella tillämpningsområden? Vad är en ”Confusion Matrix”?

Ett klassificieringsproblem handlar om att träna en modell så att den kan prediktera klasser korrekt. En confusion matrix är ett sätt att utvärdera hur många gånger en specifikt klass var korrekt predikterad eller inte. Den mäter antalet korrekta prediktioner och antalet fel (samt specifierar felet)

Tillämpningsområden: Fingeravtryck, Kamera identification.

Modeller: Support vector machine, Decision Tree, RandomForestClassifier

6. Vad är K-means modellen för något? Ge ett exempel på vad det kan tillämpas på.

K-means är en algorithm som kan hjälpa till att klustra data. Används ofta inom kundsegmentering för företag som sysslar med marknadsföring.

7. Förklara (gärna med ett exempel): Ordinal encoding, one-hot encoding, dummy variable encoding. Se mappen ”l8” på GitHub om du behöver repetition.

Ordinal encoding konverterar kategorier baserat på text med en naturlig ordning till siffror. Exempel skulle kunna vara “bra” = 0 “medel” = 1 och “dålig” = 2.

One-hot encoding skapar en binär representation av kategorierna. Se exempel nedan:

Bra	Medel	Dålig
1	0	0
0	1	0
0	0	1

Dummy Encoding är likt Ordinal encoding fast du kan ta bort en kolumn eftersom du kan utläsa den tredje kolumnen från de andra två.

8. Göran påstår att datan antingen är ”ordinal” eller ”nominal”. Julia säger att detta måste tolkas. Hon ger ett exempel med att färger såsom {röd, grön, blå} generellt sett inte har någon inbördes ordning (nominal) men om du har en röd skjorta så är du vackrast på festen (ordinal) – vem har rätt?

Det finns en poäng i det Julia säger men samtidigt är det väldigt subjektivt att säga att just en röd skjorta är vackrast. Men eftersom det alltid kommer att finnas en subjektiv tolkning av vilken färg som är snyggast på fest så är det rimligt att Julia har rätt eftersom då kan färg både vara nominal och ordinal.

9. Kolla följande video om Streamlit:
https://www.youtube.com/watch?v=ggDaRzPP7A&list=PLgzaMbMPEHEx9Als3F3sKKXexWnyEKH45&index=12

Och besvara följande fråga: - Vad är Streamlit för något och vad kan det användas till?

Streamlit är en open-source framework för att skapa data applikationer i python för ML och data science team.

6. Självutvärdering

1. Utmaningar du haft under arbetet samt hur du hanterat dem.

En utmaningen var att hantera preprocessdelen för streamlit applikationen. Med tanke på att jag valde att använda flera olika alternativ att mata in en bild blev det svårt att lösa detta.
En annan utmaning var att skriva rapporten, det märks att det var ett decenium jag senast skrev en akademisk rapport/uppsats. Jag tror jag hade lite fel tillvägagångssätt men det var en bra lärodom och förhoppningsvis bättre struktur inför nästa inlämning.

2. Vilket betyg du anser att du skall ha och varför.

Jag tycker att jag förtjänar G eftersom jag har klarat alla grundläggande moment i kursen.

3. Något du vill lyfta fram till Antonio?

Tack för denna väldigt spännande och lärorika kurs! Det känns som om att man har lärt sig väldigt mycket!

Appendix A

https://github.com/salojoakim/ML_Kunskapskontroll/tree/main

Källförteckning

Spiceworks (2024). *What is Logistic Regression?* Hämtad från Spiceworks.com

Scikit-learn Documentation. Hämtad från https://scikit-learn.org

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.