Inläsning av siffror

Ett maskininlärningsflöde med MNIST



Frida Kilby

EC Utbildning

Inläsning av siffror

202403

# Abstract

To make a workflow for Machine learning there are several steps to process. Data is going to be collected and prepared before you can choose a model to fit. How to go through these steps are presented in this report. For the chosen dataset MNIST the winning model was Support Vector Machine.

Innehållsförteckning

[Abstract 2](#_Toc161999281)

[1 Inledning 1](#_Toc161999282)

[2 Teori 2](#_Toc161999283)

[2.1 Klassificeringsmodeller 2](#_Toc161999284)

[2.1.1 Logistic Regression 2](#_Toc161999285)

[2.1.2 Support Vector Machine – Support Vector Classifier 2](#_Toc161999286)

[2.1.3 KNNeighbors 2](#_Toc161999287)

[2.2 Välja hyperparametrar i SVM modell 2](#_Toc161999288)

[2.2.1 C 2](#_Toc161999289)

[2.2.2 Gamma 2](#_Toc161999290)

[2.2.3 Kernel 3](#_Toc161999291)

[2.2.4 Degree 3](#_Toc161999292)

[3 Metod 4](#_Toc161999293)

[3.1 Datainsamling 4](#_Toc161999294)

[3.2 Utforskande av data (EDA - Explory data Analysis) 4](#_Toc161999295)

[3.3 Välja modell 4](#_Toc161999296)

[3.4 Berarbetning av data 4](#_Toc161999297)

[3.5 Trimma modellen 4](#_Toc161999298)

[3.6 Träna modellen 4](#_Toc161999299)

[3.7 Inläsning av bild och prediktering av värde 5](#_Toc161999300)

[3.8 Skapande av Streamlit app 5](#_Toc161999301)

[4 Resultat och Diskussion 5](#_Toc161999302)

[4.1 Jämförelse av modeller 5](#_Toc161999303)

[4.2 Resultat av standardisering och trimning 7](#_Toc161999304)

[4.3 Resultat för vald modell 8](#_Toc161999305)

[4.4 Inläsning av bild 2](#_Toc161999306)

[4.5 Streamlit app 3](#_Toc161999307)

[5 Slutsatser 4](#_Toc161999308)

[6 Teoretiska frågor 5](#_Toc161999309)

[7 Självutvärdering 7](#_Toc161999310)

[Källförteckning 8](#_Toc161999311)

[Referenser 8](#_Toc161999312)

[Figurförteckning 8](#_Toc161999313)

# Inledning

Genom att träna modeller på olika dataset kan sedan modellerna prediktera, läsa in nya observationer och bestämma dess värden. Det är vad som kallas för maskinlärning. Inom maskininlärning finns det regressionsmodeller och klassificeringsmodeller. Dessa används för att träna modellerna förutse data på olika sätt, regressionsmodeller används när den beroende variabeln har kontinuerliga värde medan klassificeringsmodeller används när den beroende variabeln ska delas in i två eller flera klasser. Några vanliga typer av modeller inom klassificering är Logistic Regression, Support Vector Machine samt KNNeighbors. Klassificeringsmodeller kan användas för att prediktera, läsa in exempelvis handskrivna siffror, observationers värde. Maskininlärningsmodeller som ska utföra denna typ av inläsning tränas ofta genom att använda det kända datasetet MNIST.

Syftet med denna rapport är att uppvisa kunskaper inom maskininlärning, för att uppfylla syftet så kommer följande frågeställningar att besvaras:

1. Vilken av tre på förhand valda modeller tränas bäst på datasetet MNIST
2. Hur skapas ett komplett ML-flöde från start till slut

# Teori

## Klassificeringsmodeller

Det finns olika tillämpningsområden för klassificeringsmodeller, kortfattat så används klassificering när modellen ska tränas på att dela upp data i två eller fler klasser. Så som SANT/FALSKT, JA/NEJ, MAN/KVINNA vilka alla har två möjliga klasser och kallas därför binära klassificeringsproblem.

Vanligen används klassificeringsmodeller för att svara på frågor så som:  
- kommer den här kunden att återkomma  
- är e-postmeddelandet skräppost  
- är det en man eller kvinna  
- sannolikheten för att det ena av två saker kommer att inträffa

När modellen ska dela in data i fler klasser än två kallas det för multiklassificering.

### Logistic Regression

Logistic Regression tillhör de linjära modellerna, den används trots sitt namn för att klassificera data. Precis som andra linjära modeller kan denna klassas med l1 och l2, som kallas C och gamma. (Géron, 2019)

### Support Vector Machine – Support Vector Classifier

Går att använda för såväl regressionsmodeller som klassificeringsmodeller. Modellen separerar klasser i hyperplanet med så bred gata som möjligt mellan variablerna, den transformerar data till högre dimensioner för att enklare kunna hitta de linjära separationerna.

### KNNeighbors

KNNeighbors tillhör neighbors-nbaserade inlärningsmetoder, vilken går ut på att hitta ett bestämt antal träningsvariabler nära observationen och förutse klassen från detta. Modellen kan hantera så väl NumPy arrayer och scipy-sparse matriser och går att använda både för klassificering och regressionsproblem. (1.6. Nearest Neighbors - scikit-learn 1.4.1 documentation, 2024)

## Välja hyperparametrar i SVM modell

En hyperparameter heter i scikit-biblioteket parameter och påverkar datasetet till skillnad mot en parameter som påverkar själva modellen och kallas attribut. Om hyperparametrar sätts för högt och påverkar data för mycket är det lätt hänt att modellen blir överanpassad på träningsdata.

### C

C är en parameter för regularisering som påverkar flexibiliteten och över- eller underanpassad, delar data med följsamt. Om värdet är lågt får vi en bred gata men även en låg träffsäkerhet på observationernas klassning, medan om värdet är högt arbetar modellen hårt för att ge oss en smal gata och att rama in alla variabler korrekt. (Understanding Parameters of SVM, 2024)

### Gamma

Gamma påverkar också flexibiliteten samt över- och underanpassning. Parametern gör att data ramas in mer följsamt. Detta genom att med värdet ange hur mycket träningsexemplet påverkar, ju större gamma är desto fler närliggande variabler påverkas. (Understanding Parameters of SVM, 2024)

### Kernel

Kernel anger med vilken matematiskfunktions som klasserna ska delas in. Är kernel satt till linjär används linjer för att dela in data, är Kernel satt till RBF (Radial Based Function) delas data istället in med hjälp av en formel som skapar rundade inramningar och för kernel poly används en formel som ger en inramning av polynomialaekvationer. (1.4. Support Vector Machines - scikit-learn 1.4.1 documentation, 2024)

### Degree

Degree används bara av poly-kernel, parametern styr hur flexibel modellen ska vara i sina klassificeringar. Om degree är satt till 1, arbetar poly-kernel på ett linjärt sätt. (Understanding Parameters of SVM, 2024)

En bild som visar text, Teckensnitt, skärmbild, linje

Automatiskt genererad beskrivning

En bild som visar text, skärmbild

Automatiskt genererad beskrivning

Figur 1 skillnader mellan olika kernel och dess funktioner. (1.4. Support Vector Machines - scikit-learn 1.4.1 documentation, 2024)

# Metod

Biblioteket scikit-learn är ett open source bibliotek som hela tiden fylls på med nya funktioner genom att koden är publik och alla kan komma åt att utveckla och använda den. En stor fördel i scikit-learn är att det är konsekvent vilket innebär att flertalet kommandon och namngivning för exempelvis metoder och attribut är återkommande genom hela biblioteket. En annan fördel är standardvärden vilket gör att flertalet av parametrarnas värden är satta som standard så det är enkelt att komma igång och träna modellerna utan några större insatser.

## Datainsamling

För att träna tre modeller och utvärdera dessa på dataset MNIST kommer ett maskininlärningsflöde att upprättats. Först läses data in från setet med hjälp av fetch opennml därefter delades datasetet i tränings och test set. Utöver det delades tränings-set in i tränings- och validerings set med hjälp av train\_test\_split funktionen.

## Utforskande av data (EDA - Explory data Analysis)

För att utforska data gjordes en så kallad EDA (Explory Data Analysis). Datasetets form skrevs ut tillsammans med dess information, hur första värdet ser ut och med hjälp av pyplot skrevs några av datasetets olika siffror ut. Siffrorna är i dataset är sparade i numpy ndarray.

## Välja modell

De tre modeller som valdes att träna och utvärdera data på var Logistic Regression, Support Vector Machines och KNNeighbors. Alla tre modellerna testades med standardvärden och utvärderades med hjälp av accuracy\_score, denna funktion räknar ut modellernas precision. Till dessa användes tränings- och validerings-setet av data. Utöver detta skapades även en confusion matrix per modell som visar hur ofta modellerna predikterar korrekt, felaktigt och för vilka värden.

## Berarbetning av data

Support Vector Machine var den modell som fick högst precision i tidigare steg och således valdes denna modell för att arbeta vidare med. För att modeller av typen Support Vector Macinge ska fungera optimalt så ska dessa tränas med standardiserade data. Detta gjordes med Standardscaler.

## Trimma modellen

Ett sätt att trimma modellen är att göra en GridSerachCV på modellen, det som genomförs då är att alla hyper-parametrar som angetts gås igenom och modellen tränar om sig själv med de bästa hyper-parametrarna. Detta görs för att modellen ska bli ytterligare något bättre än vad den är i sitt standardutförande. För att kunna utföra detta steg inom en lämplig tid användes enbart en del av tränings- och validerings-setet till detta, 5000 i tränings-set och 1000 i test-set.

## Träna modellen

Efter att hyperparameterar valts med hjälp av GridSearch tränades modellen om på det kompletta tränings-setet som även inkluderade validering-setet och kontrollerades med test-setet. Detta till skillnad mot tidigare träning som skett på tränings- och validerings-setet och kontrollerats med validerings-setet eller delar av dessa.

## Inläsning av bild och prediktering av värde

För att kunna läsa in egna bilder lades biblioteket cv2 till. Med hjälp av det biblioteket skapades en funktion som läste in och transformerade egna bilder. Bilderna transformerades enligt stegen:  
- gråskala  
- tröskelvärde för att göra bilden svart-vit  
- ändring av storlek till 28 x 28 pixlar  
- inventering av färger  
- kollapsa array till en dimension  
- gjorde om formen på array för att passa datasetet  
- standardiserade bilden med hjälp av sparad StandardScaler-modell

Därefter kördes modell för att förutse vad bilden visade för värde.

## Skapande av Streamlit app

Streamlit är ett enkelt och användarvänligt sätt att kunna presentera skript och göra applikationer. En python-fil innehållandes kod för inläsning av bilder samt prediktering med den tränade modellen skapades för att kunna köra med streamlit lokalt på datorn.

# Resultat och Diskussion

## Jämförelse av modeller

Support Vector Machine valdes då precisionen för modellen var den högsta av de tre tränades modellerna när de tränades på tränings- och validerings-set med standardvärden. Det var dock väldigt jämt mellan KNNeighbors och Support Vector Machine då det enbart skiljde 0,0086 i accuarcy score.

Resultat från modellernas jämförelse, vid oskalade modeller med standardinställning:

|  |  |
| --- | --- |
| **Accuracy score** | |
| Logistic Regression | 0,9186 |
| Support Vector Machine | 0,9788 |
| KNNeighbors | 0,9702 |

Tabell 1: Accuracy score (accuracy\_score()) för de tre valda modellerna.

En bild som visar skärmbild, kvadrat, Färggrann, pixel

Automatiskt genererad beskrivning

Figur 2 Confusion matrix från Logistic Regression

En bild som visar skärmbild, kvadrat, text, diagram

Automatiskt genererad beskrivning

Figur 3 Confusion matrix från Support Vector Machine

En bild som visar skärmbild, kvadrat, Färggrann, pixel

Automatiskt genererad beskrivning

Figur 4 Confusion matrix från KNNeighbors

## Resultat av standardisering och trimning

Då modeller som tillhör Support Vector Machine arbetar optimalt på standardiserade värden så standardiserades tränings- och valideringssetet. En ny träning gjordes därefter på standardiserade data, precisionen för denna blev något sämre än med grundvärden. 0,9632 mot tidigare 0,9788.

En bild som visar text, skärmbild, Teckensnitt, nummer

Automatiskt genererad beskrivning

Figur 5 Resultat av vald modell före införande av hyperparametrar

För att hitta bästa hyperparametrar genomfördes en GridSearch på datan. Men då detta är tidskrävande gjordes detta enbart på 5000 värden med ett test-set på 1000 värden. Hyperparametrar som enligt jämförelsen ska väljas är C=0.01, gamma=1, kernel=poly och degree=3. Detta ger en tränad modell med precision 0,97, vilket är något sämre än modellens värde för icke-standardiserade värden och grundinställningar.

En bild som visar text, skärmbild, Teckensnitt, programvara

Automatiskt genererad beskrivning

Figur 6 Resultat av Gridsearch för Support Vector Machine

En bild som visar text, skärmbild, linje, Teckensnitt

Automatiskt genererad beskrivning

Figur 7 Resultat av GridSearch för enbart degree och kernel=poly

En bild som visar text, skärmbild, Teckensnitt, nummer

Automatiskt genererad beskrivning

Figur 8 Resultat av modellträning med valda hyperparametrar

## Resultat för vald modell

Modell med valda hyperparametrar tränades om på tränings-set inkl. validerings-set. Detta gav en precision på 0,9788, samma som i initiala testet för att välja modell.

En bild som visar text, skärmbild, Teckensnitt, linje

Automatiskt genererad beskrivning

Figur 9 Resultat av omtränad modell

En bild som visar skärmbild, kvadrat, Färggrann, diagram

Automatiskt genererad beskrivning

Figur 11 Confusion Matrix för den omtränade modellen

En bild som visar text, skärmbild, nummer, Teckensnitt

Automatiskt genererad beskrivning

Figur 12 Classificationsrapport för den omtränade modellen

## Inläsning av bild

Prediktering av egna bilder gjordes med hjälp av flödet för transformering.   
Siffrorna predikterades med modellen som tränats om med hyperparameterar, denna modell hade sparats som en joblib fil för att undvika att träna modellen vid varje körning.

En bild som visar skiss, Rektangel, diagram, design

Automatiskt genererad beskrivning

Figur 13 Resultat vid inläsning av egna siffror skapade i paint, samt siffrorna skapade i paint

En bild som visar diagram, skiss, Rektangel, text

Automatiskt genererad beskrivning

Figur 14 Resultat vid inläsning av egna siffror ritade med tuschpenna, samt siffrorna skapade med tuschpenna

## Streamlit app

En bild som visar text, skärmbild, programvara, Multimedieprogram

Automatiskt genererad beskrivning

Figur 15 Urklipp från Streamlit app som predikterat inläst siffra 3 lokalt på datorn

# Slutsatser

För prediktering av data från setet MNIST var Support Vector Machine den bättre modellen att välja när denna tillsammans med KNNeighbors och Logistic Regression testades med standardparametrar. Med hjälp av hyperparametrar lyckades modellen trimmas till ett testresultat på 0,9788 på det kompletta tränings-setet. Detta är samma resultat som modellen fick med standardparameterar för del av träningsdata. Av nyfikenhet kontrollerades vad modellen fått för precision med standardinställningar på det kompletta standardiserade setet, vilket hade blivit 0,93. Resultat är ett bra resultat med tanke på att modellen som används är av typen övervakad inlärningsmodell.

Den trimmade modellen klarade av att prediktera nya siffor men med sämre träffsäkerhet. Det var av yttersta vikt att siffrorna påminde om de som MNIST innehöll. Att siffrorna hade väldigt vit bakgrund, god kontrast och var skrivna på samma sätt och med en tjock penna. Det går säkert att fortsätta träna modellen på andra typer av siffror över tid.

Det kompletta maskininlärningsflödet som skapades för den här rapporten innehöll stegen

* Datainsamling
* Utforskande av data
* Uppdelning i tränings-, validerings- och test-set
* Bearbetning av data
* Träning och utvärdering av modeller
* Beroende på val av modell kompletterande bearbetning av data
* Trimning av modell
* Träning av modell på hela tränings-setet
* Utvärdering av modellen
* Gör predikteringar

Fortsättning

Är det önskvärt att inläsningen och predikteringen av egna siffror ska bli bättre är det rekommenderat att fundera vidare på neurala nätverk eller andra modeller för inlärning, Detta då dessa är bättre på att lära sig identifiera mer komplexa mönster. Väljs även en modell som tillhör typen oövervakade modeller och kommer således att utvecklas utan övervakning med tiden.

# Teoretiska frågor

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

*Träningsdelen används för att träna en eller flera modeller på.  
Valideringsdelen används för att utvärdera vilken modell som fungerar bäst samt för att finjustera hyper-parametrar med. Valideringsdelen av data skapas från träningsdata.  
Testdelen används för att utvärdera hur bra din valda modell blev, viktigt är att träna modellen på fullt träning- och valideringsdata innan man utvärderar på test-set.*

1. Julia delar upp sin data i träning och test. På träningsdata 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 ”validerings-dataset”?

*Då kan Julia använda sig av krossvalidering, en metod för detta är K-fold CV där träningsdata delas k gånger. Modellen tränas sedan på data k minus en gång och resultatet valideras på sista portionen data som inte användes för träning. Julia kan sedan använda medelvärdet av sina K-folds för att utvärdera vilken modell som ska användas.*

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

*Ett regressionsproblem är när vi arbetar med en modell som ska förutse värden, där den beroende variabeln har kontinuerliga värden. Exempel på regressionsproblem kan vara inkomst mot ålder att förutse kommande inkomst. Några modeller inom regressionsproblem är Linear Regression (Lasso Regression, Ridge Regression) Random Forest regression samt Support Vector Machines (SVR).*

1. Hur kan du tolka RMSE och vad används det till:

*Root mean squared error är roten ut medelvärdet för avståndet (absolutbeloppet) mellan de predikterade värdena (y tak) och faktiska värdena (y). Vilket används till att förklara hur bra eller dålig en modell är. Root Mean Squared Error ska vara så liten som möjligt. Och används även till att visa hur bra test-data har blivit predikterad.*

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

*Ett klassificeringsproblem är när data/observationer ska delas in i olika klasser, exempelvis ja eller nej, blå eller röd vilka är binära problem, men modellerna kan även användas på multiklassproblem.  
Några typer av modeller som är klassificeringsmodeller är Logistic Regression, Support Vector Machines (SVC) samt Random Forest Classification, KNN.*

*En confusionmatrix är antingen enbart siffror eller en figur i form av en fyrkant/matris som visar hur många korrekta och felaktiga prediktioner som modellen skapade på ett förståeligt sätt. Detta genom att visa Sanna Negativa, Falska Positiva, Falska Negativa och Sanna Positiva. Matrisen har korrekta prediktioner längs med diagonalen och övriga delar ovan eller under diagonalen.   
  
Några tillämpningar för klassificeringsmodeller är SPAM filter, kommer spelaren/kunden att spela igen (churna) eller liknande frågeställningar.*

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

*K-means modellen är när vi redan har bestämt oss för hur många grupper som vi vill att ett klassifieringsproblem ska delas upp i. K står för hur många och modellen är en unsuperivsed inlärningsalgorithm (clustering). Grupperna är för oss okända innan vi kör modellen, så modellen själv väljer hur data ska delas in i grupper. Denna modell kan användas till marknadsföring, man kan se vilka som handlar vad, målstyra kunder eller optimera leveransturer.*

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

*Alla tre är tekniker för att genomföra förberedande transformationer. Detta för att kunna använda kategoriska data i exempelvis linjära regressioner som förväntar sig numeriska data.*

*Ordinal encodning är när dummys för klasser har ökande värde, när man exempelvis vill favorisera något eller när det redan finns inbördes ordning*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***Klass*** | ***dummy*** |  | ***värde*** | ***dummy*** |  | ***värde*** | ***dummy*** |
| *Röd* | *1* |  | *100* | *1* |  | *IG* | *1* |
| *Grön* | *2* |  | *200* | *2* |  | *G* | *2* |
| *Blå* | *3* |  | *300* | *3* |  | *VG* | *3* |

*One-hot encodning är när alla klasser har en dummy, att antingen är det röd, grön eller blå.*

|  |  |  |  |
| --- | --- | --- | --- |
| ***Klass*** | ***om\_röd*** | ***om\_grön*** | ***om\_blå*** |
| *Röd* | *1* | *0* | *0* |
| *Grön* | *0* | *1* | *0* |
| *Blå* | *0* | *0* | *1* |

*Dummy Variable är när vi kan droppa en dummy för en klass, att antingen är det eller är det inte. Exempelvis om det inte är röd eller grön så är den blå. Detta används oftast för linjär regression.*

|  |  |  |
| --- | --- | --- |
| ***Klass*** | ***om\_röd*** | ***om\_grön*** |
| *Röd* | *1* | *0* |
| *Grön* | *0* | *1* |
| *Blå* | *0* | *0* |

1. 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?

*Julia har rätt i frågan, då det i vissa fall är en tolkningsfråga, i Exemplet med färger så kan färger finnas utan inbördes ordning (nomial), men om det handlar om favoritfärg så är det helt klart en ordinal data.*

1. Kolla följande video om Streamlit: https://www.youtube.com/watch?v=ggDa-RzPP7A&list=PLgzaMbMPEHEx9Als3F3sKKXexWnyEKH45&index=12 Och besvara följande fråga: - Vad är Streamlit för något och vad kan det användas till?   
   *Ett enkelt sätt att dela med sig av sina script exempelvis gällande machine learning är att dela dessa via en app. Genom streamlit kan man göra detta genom att dela en länk till streamlit som läser kod från github så att andra kan testa dina modeller och göra prediktioner.*

# Självutvärdering

1. Utmaningar du haft under arbetet samt hur du hanterat dem.   
   *Utmaningarna med kursen och kunskapskontrollen har i vanlig ordning varit tid. Det är ett väldigt intressant område och stort område. Boken har tyvärr blivit bortprioriterad att läsa i, men kommer användas som källa och läsning framöver när det är kunskap som saknas. Det var rätt enkelt att komma igång med programmeringen förutsatt att det inte blev så länge mellan tillfällena. Tröskeln i förståelsen var hög initialt men nu känns många delar solklart.*
2. Vilket betyg du anser att du skall ha och varför.   
   *Ett VG är alltid kul men jag skulle bli precis lika glad för ett G.*
3. Något du vill lyfta fram till Antonio?

*Generellt sätt en väldigt kul kurs, härligt att vara tillbaka i gruppdiskussioner och ett väldigt bra sätt att lära sig på med instuderingsfrågor och att diskutera dessa. Det gick bra mycket enklare att diskutera frågorna än att diskutera sin egen rapport.*

# Källförteckning

# Referenser

*1.4. Support Vector Machines - scikit-learn 1.4.1 documentation*. (den 21 03 2024). Hämtat från scikit-learn: https://scikit-learn.org/stable/modules/svm.html

*1.6. Nearest Neighbors - scikit-learn 1.4.1 documentation*. (den 21 03 2024). Hämtat från scikit-learn: https://scikit-learn.org/stable/modules/neighbors.html#classification

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow.* Sebastopol: O’Reilly Media,.

*Understanding Parameters of SVM*. (den 21 03 2024). Hämtat från kaggle: https://www.kaggle.com/code/gorkemgunay/understanding-parameters-of-svm

# Figurförteckning

[Figur 1 skillnader mellan olika kernel och dess funktioner. (1.4. Support Vector Machines - scikit-learn 1.4.1 documentation, 2024) 3](#_Toc161949094)

[Figur 2 Confusion matrix från Logistic Regression 5](#_Toc161949095)

[Figur 3 Confusion matrix från Support Vector Machine 6](#_Toc161949096)

[Figur 4 Confusion matrix från KNNeighbors 6](#_Toc161949097)

[Figur 5 Resultat av vald modell före införande av hyperparametrar 7](#_Toc161949098)

[Figur 6 Resultat av Gridsearch för Support Vector Machine 7](#_Toc161949099)

[Figur 7 Resultat av GridSearch för enbart degree och kernel=poly 7](#_Toc161949100)

[Figur 8 Resultat av modellträning med valda hyperparametrar 8](#_Toc161949101)

[Figur 9 Resultat av omtränad modell 8](#_Toc161949102)

[Figur 11 Confusion Matrix för den omtränade modellen 8](#_Toc161949103)

[Figur 12 Classificationsrapport för den omtränade modellen 8](#_Toc161949104)

[Figur 13 Resultat vid inläsning av egna siffror skapade i paint, samt siffrorna skapade i paint 2](#_Toc161949105)

[Figur 14 Resultat vid inläsning av egna siffror ritade med tuschpenna, samt siffrorna skapade med tuschpenna 2](#_Toc161949106)

[Figur 15 Urklipp från Streamlit app som predikterat inläst siffra 3 3](#_Toc161949107)