Machine Learing

The art of image recognition



Leonardo Sjöberg

EC Utbildning

Data Science

2024-03

# Abstract

This work focuses on image recognition with help of the MNIST-dataset. The primary objective of this project is to find the best fitting model for the dataset, as well of predicting images of own handwritten numbers. The research explores different models such as the Support Vector Machine and Random Forest Classifier. It will also search the best hyperparamaters of a model with help of GridSearchCV, and it’ll use different evaluation metrics to classify the best model of a subset of models.

Innehållsförteckning

[Abstract 2](#_Toc162004903)

[1 Inledning 1](#_Toc162004904)

[1.1 Syfte och frågeställning 1](#_Toc162004905)

[2 Teori 2](#_Toc162004906)

[2.1 Utvärderingsmetriker 2](#_Toc162004907)

[2.2 Modeller 3](#_Toc162004908)

[2.2.1 Decision Tree 3](#_Toc162004909)

[2.2.2 Random Forest 3](#_Toc162004910)

[2.2.3 Extra Trees Classifier 3](#_Toc162004911)

[2.2.4 K-Nearest Neighbors 3](#_Toc162004912)

[2.2.5 Support Vector Machine 4](#_Toc162004913)

[2.3 Hyperparameter justering 4](#_Toc162004914)

[2.3.1 GridSearchCV 4](#_Toc162004915)

[3 Metod 5](#_Toc162004916)

[3.1 Data 5](#_Toc162004917)

[3.1.1 MNIST 5](#_Toc162004918)

[3.1.2 Egna Bilder 5](#_Toc162004919)

[3.2 Modelval 5](#_Toc162004920)

[3.3 Hyperparameter 5](#_Toc162004921)

[3.4 Evaluering 6](#_Toc162004922)

[4 Resultat och Diskussion 7](#_Toc162004923)

[4.1 Resultat, Evaluering och Test 7](#_Toc162004924)

[4.1.1 Modelval 7](#_Toc162004925)

[4.1.2 Evaluering 7](#_Toc162004926)

[4.1.3 Test 11](#_Toc162004927)

[4.1.4 Egna Bilder 12](#_Toc162004928)

[4.2 Diskussion 13](#_Toc162004929)

[4.2.1 Modelval 13](#_Toc162004930)

[4.2.2 Evaluering 13](#_Toc162004931)

[4.2.3 Test 14](#_Toc162004932)

[4.2.4 Egna Bilder 14](#_Toc162004933)

[5 Slutsatser 15](#_Toc162004934)

[6 Teoretiska frågor 16](#_Toc162004935)

[7 Självutvärdering 17](#_Toc162004936)

[Appendix A 18](#_Toc162004937)

[Källförteckning 19](#_Toc162004938)

# Inledning

Artificiell intelligens och Machine learning har över tid blivit ett alltmer större ämne och en väldigt viktig del för många institutioner. Det har revolutionerat hur datorprogram fungerar och interagerar med världen. Ett område där Machine Learning och artificiell intelligens har haft stor inverkan är bildigenkänning. Med hjälp av dessa tekniker kan datorprogram identifiera och kategorisera bilder med en noggrannhet som efterliknar oss människor och vissa fall bättre.

Ett exempel på användning av bildigenkänning med hjälp av Machine Learning är hur man använder MNIST-datasetet för att förutsäga vilket nummer som representeras på en bild. MNIST-datasetet består av handskrivna siffror från ett till nio och datasetet innehåller 70 000 olika bilder. Många modeller har använts för att prediktera siffror med hjälp av MNIST-datasetet, men finns det en modell som är bäst för datasetet?

Modellerna tränas och testa oftast på data som efterliknar verkligheten. MNIST-datasetet är ett välkänt och fint dataset för nybörjare att använda och lära sig från, men hur presterar modellerna när man ger dem bilder med siffror som är tagna i stunden. Kommer modellerna att prestera lika bra eller kommer de ge stora problem för prediktionerna?

## Syfte och frågeställning

Syftet med denna rapport är att undersöka hur Machine Learning kan användas för att förutspå siffror på bilder, samt försöka hitta den bästa modellen från ett urval av modeller. För att uppnå detta kommer följande frågeställning att besvaras:

* Vilken modell förutsäger bäst ut av ett urval av modeller?
* Kan modellen prediktera mina handskrivna bilder?

Detta arbete kommer att fokusera på området inom bildigenkänning med hjälp av Machine Learning och kommer inte täcka andra tillämpningar av artificiell intelligens eller Machine Learning. Detta gör arbetet mer hanterbart och fokuserat.

Projektet kommer att inledas med att ladda in data och utforska data för saknade värden och annat som är viktigt för att kunna använda den till träning och testning av modeller. Sedan kommer modellerna att finjusteras och förbättras genom att ändra deras hyperparametrar. Efter det kommer modellerna evalueras med eventuellt sina bästa hyperparametrar och den bästa modellen kommer att testas med testdata och egna handskrivna bilder.

# Teori

Denna del av rapporten kommer gå igenom teori som är nödvändig för att förstå modellerna och vad utvärderingsmetrikerna betyder och hur vi kan använda dem för att evaluera våra modellers styrka.

## Utvärderingsmetriker

Utvärderingsmetriker har en väldigt stor roll inom Machine Learning. Det ger oss ett sett att mäta hur väl vår modell presterar samt hjälper oss förstå dess styrkor och svagheter. Inom bildigenkänning med målet att klassificera korrekta bilder kan vi använda metriker som Accuracy, Precission, Recall, F1-score och Confusion Matrix för att bedöma en modells prestanda. Genom att utvärdera modeller med dessa metriker kan vi göra informerade beslut om hur vi kan förbättra modellen, samt få en bättre inblick på vilka tekniker vi ska använda för att bäst kunna uppnå målen vi söker. Utan dessa utvärderingsmetriker skulle vårt arbete att bli betydligt mycket svårare och vi skulle få rent av gissa fram modellernas prestanda. Därför är dessa utvärderingsmetriker en grundläggande del av alla maskininlärningsprojekt.

Här är mer ingående på de olika utvärderingsmetrikerna.

* Confusion Matrix enligt Géron är en central utvärderingsmetrik. Den är en specifik tabelllayout som ger en mer detaljerad uppdelning av en modells prestanda än de flesta andra metriker. Varje rad representerar den faktiska klassen medan kolumnerna representerar den predikterade klassen. Man kan dela upp Confusion Matrix i fyra olika komponenter: sanna positiva, falska positiva, sanna negativa och falska negativa (2019, p.92). I detta projekt kommer Confusion Matrix att vara en viktig metrik, eftersom den ger en mer detaljerad bild av modellens prestanda. Confusion Matrix visar totala antalet korrekta och felaktiga predictioner vilket gör det möjligt att se exakt var modellen gör misstag, vilket kan vara mycket värdefullt för att förbättra dess prestanda.
* Accuracy enligt Géron är den mest vanliga utvärderingsmetriken och är förhållandet mellan det som är korrekt predikterat och det totala antalet prediktioner (2019, p.4). Denna utvärderingsmetrik kommer att vara viktig för detta projekt då vi vill ha så många rätt över lag och så få fel som möjligt.
* Precision, enligt Géron är en annan viktig utvärderingsmetrik inom maskininlärning. Den definieras som förhållandet mellan de korrekt predikterade positiva fallen och det totala antalet positiva prediktioner. Detta innebär att precision mäter hur många av de positiva prediktionerna som faktiskt var korrekta. (2019, p.93). I detta projekt kommer Precision att vara en viktig metrik, eftersom vi vill minimera antalet falska positiva resultat, det vill säga de fall där modellen felaktigt identifierar en bild som ett visst nummer.
* Recall är en ytterligare viktig utvärderingsmetrik som enligt Géron. Den definieras som förhållandet mellan de korrekt predikterade positiva fallen och det totala antalet faktiska positiva fall. Detta innebär att Recall mäter hur många av de faktiska positiva fallen som korrekt identifierades av modellen. (2019, p.94). I detta projekt kommer Recall att vara en viktig metrik, eftersom vi vill minimera antalet falska negativa resultat, det vill säga de fall där modellen felaktigt missar att identifiera en bild som ett visst nummer.
* F1-score enligt Géron är en annan central utvärderingsmetrik. Den definieras som det harmoniska medelvärdet av precision och Recall, vilket innebär att den tar hänsyn till både falska positiva och falska negativa resultat. Man kan säga att det är en viktad kombination av de två metrikerna. En hög F1-poäng indikerar att både Precisionen och Recall är höga, men en låg Recall och hög precision ger ett lågt F1-score. (2019, p.4). I detta projekt kommer F1-poängen att vara en viktig metrik, eftersom vi vill ha en bra balans mellan att Precision och Recall.

## Modeller

När det kommer till bildigenkänning så kan man välja modeller som är bra med att hantera klassifikations problem då vår data är indelad i flera olika klasser, från noll till nio. Detta är de olika klassifikations modellerna som har testats.

### Decision Tree

Decision Tree Classifier är en kraftfull modell som kan användas för både klassificering problem och för regressionsproblem. Modellen kan beskrivas som ett flödesschema där modellen skapar en inre nod som representerar en fråga som besvaras vilket skapar en gren som går till en ny nod, detta fortsätter tills man träffar den sista noden (”bladnoden”) som representerar klassificeringen eller prediktionen. (Géron, 2019, p. 179) En nackdel med denna modell är att den kan enkelt överanpassa sig på datan och ge dåliga resultat för när den är ute i produktion. Fördel med den är att den kan vara snabb att träna upp.

### Random Forest

Random Forest är väldigt kraftfull modell som kan användas för både klassificering och regressionsproblem. Modellen skapar flera Decision Trees som arbetar tillsammans. I stället för att hitta den bästa featuren när den delar på en nod, letar den efter den bästa featuren mellan en slumpmässigt mängd features, därför namnet Random Forest. (Géron, 2019, p.199) Fördelen med denna modell är att den sänker variansen på modellen för att ge bättre prestanda, nackdelen kan bli väldigt långsam när många träd ska användas med ett stort dataset.

### Extra Trees Classifier

Extra Trees Classifier fungerar likadant som Random Forest men gör processen ännu mer slumpmässigt vilket sänker variansen ännu mer för modeller och är mycket snabbare att träna. (Géron, 2019, p.200)

### K-Nearest Neighbors

K-Nearest Neighbors är en simpel modell som är lätt att implementera. Hur modellen fungerar är att den antar att klasserna i datan är grupperat och ligger nära varandra. K är antalet datapunkter som krävs för att skapa ett område för klassen. Om K är 3 så krävs det t. ex. tre röda punkter för att skapa ett område för den klassen och hamnar en ny data punkt inom området så kommer vi att prediktera det värdet. Fördelen med modellen är att den är väldigt simpel och lätt att förstå. Nackdelen är att den blir datan ökar. (Harrison. 2018)

### Support Vector Machine

Support Vector Machine är en väldigt populär och kraftfull Modell inom Maskin inlärning. Support Vector Machine delar tar datan och sätter gränser runt om datan och skapar områden som tillhör de olika klasserna. Modellen predikterar genom att välja det området med klassen som datan hamnar inom. Fördelen med denna modell är att den är väldigt kraftfull och bra för både klassifikations och regressions problem. En stor nackdel för modellen är att den blir väldigt långsam att träna om man har ett väldigt stort dataset. (Géron, 2019, p.155)

## Hyperparameter justering

En stor del av att förbättra sin modell kan göras genom att ändra på hyperparametrar. Genom att ändra på dessa parametrar kan man till exempel straffa modellen när den predikterar fel och då få den att lära sig bättre av dessa fel. Ett sätt att hitta bra och passande hyperparametrar är genom att använda sig av GridSearchCV som letar efter de bästa värdena för parametrarna som man angivit.

### GridSearchCV

GridSearchCV är ett väldigt viktigt och kraftfullt verktyg för att hitta rätt hyperparametrar som passar datan och modellen bäst. GridSearchCV fungerar genom att man tar modellerna och hyperparametrarna och testar alla olika kombinationer av värden som man vill att modellen ska testa. Seden delar den upp datan i n olika delmängder, beroende på vad min vill ha och tränar modellen med de olika delmängderna av datan förutom en, där den testar modellen på den delmängden av datan och gör så n gånger för alla olika delmängder av datan. Fördelen med att använda GridSearchCV är att modellens prestanda ökar väldigt drastiskt. Nackdelen är att ju mer data och hyperparametrar du vill testa, desto längre tid tar det för att hitta den bästa kombinationen, och det kan öka väldigt i tiden det tar. (Géron, 2019, p. 79)

z

# Metod

Denna del av rapporten kommer att gå igenom hur svaren till frågeställningarna har åstadkommits. Vad som kommer ingå i delen är hur datan har framkallats och vad den innehåller, vilka modeller som har valts samt justering av hyperparametrar för att öka modellerna prestanda och sedan evaluera den bästa modellen med olika metriker samt egna bilder.

## Data

Här presenteras hur datan till modellerna har framkallats och förändrade för att passa modellerna så att de kan göra en prediktion.

### MNIST

Datan som har använts för att träna och testa modellerna kommer från OpenML API genom sklearns fetch\_openml function. Datasetet heter MNIST-dataset (LeCun, Cortes, Burges, 2014). Detta dataset innehåller 70 000 bilder på olika handskrivna siffror från ett till nio. Detta dataset är ett mycket välkänt dataset inom Maskin inlärning området och har använts av många. Anledningen till att många använder datasetet är på grund av dess fina kvalitet. Det har inga saknade värden, det har en balanserad distribution mellan siffrorna, det är enkelt att tyda siffrorna på bilderna och där är väldigt få outliers i datasetet.

Bilderna är gråskalade och har dimensionen 28x28 vilket ger bilderna 784 olika pixlar och features för modellerna att använda för att träna på. Vi delar upp datasetet i tre delar, en träningsdel som kommer att vara 68% av all data, sedan validering data på 15% för utvärdera modellerna och utveckla deras prestanda, och till slut kommer de sista 17% att vara testdata för att se hur de finjusterade modellerna fungerar på osedda data. Datasetet är uppdelat så att de sista 10 000 av bilderna är till för att testa medan de andra 60 000 är för att träna modellen, vi delar sedan upp tränings datan till en ny tränings del och en validerings del.

### Egna Bilder

Det viktiga när egna bilder använd är att transformera om dem så att de liknar de som redan har använts för att träna upp modellerna. Bilderna måsta vara gråskalade sedan måste de ändra storleken på bilden till 1x784 för att modellen ska kunna använda alla pixlar som feature och kunna prediktera ett värde från bilden. Om bilden kommer fram med siffrorna som vit och bakgrunden som svart så måste bilderna inverteras för att det ska bli rätt.

## Modelval

Med vårt data set som vi har så bör vi hantera detta problem som ett multiclass klassifikations problem, då vi har flera olika klasser som vi försöker identifiera. För att hitta några bra modeller som är värda att utveckla på tar vi några klassifikationsmodeller, tränar upp dem med tränings data på deras standard hyperparametrar och sedan utvärderar dem på validerings data för att se vilka modeller som visar bäst prestanda. För att hitta dem modellerna så används Accuracy som utvärderingsmetrik för att se vilka modeller som presterar bäst. De modeller som har en Accuracy över 95% kommer gå vidare för att hitta bättre hyperparametrar.

## Hyperparameter

För att utveckla modellerna och öka deras prestanda kommer GridSearchCV att användas för att hitta hyperparametrar som gör modellerna bättre. Alla hyperparametrar kommer inte att ändras då det tiden för att träna alla olika varianter av modeller hade tagit alldeles för mycket tid. För Extra Trees Classifier och Random Forest Classifier så kommer samma hyperparametrar att ändras för att hitta en bra passande modell. Dem hyperparametrarna kommer att vara n\_estimators, criterion, max\_depth, min\_samples\_split, min\_samples\_leaf. För K-Neighbors Classifier kommer hyperparametrarna n\_neighbors, weights, algorithm, leaf\_size att ändras för att för att utveckla modellens prestanda. För SupportVectorMachine kommer hyperparametrarna C, kernel och gamma att ändras. Varje modell kommer att gå igenom GridSearchCV tre gånger för att ändra och hitta den bästa hyperparametern, förutom Support Vector Machine då den tar väldigt lång tid för att hitta passande parametrar.

Tabellen 3.1 visar Suppor Vector Machines bästa kombinationen mellan hyperparameter värden som har tagits fram med hjälp av GridSearchCV.

## Evaluering

Evaluering med alla metriker

När rätt parametrar är hittade för modellerna kommer vi nu att utvärdera dem på validering data och se vilken modell som är bäst av dem alla. För att utvärdera modellerna så kommer vi använda oss av utvärderingsmetrikerna Accuracy, Precision, Recall, F1-score och Confusion Matrix för att se hur modellerna presterar på datan. Den modell som har högst procentsats mellan metrikerna kommer att vara den utvalda modellen för att bli testad på testdata. Efter att den modellen är utvald kommer den att testas på egna handskrivna bilder.

|  |  |  |
| --- | --- | --- |
| **Hyperparameter** | **Värden** | **Bästa värdet** |
| C | 1.0, 5.0 | 5.0 |
| Kernel | RBF, Poly, Sigmoid | RBF |
| Gamma | Scale, Auot | Scale |

Table 3.1. exempel på hyperparametrar och värden med resultat på det bästa värdet på en Support Vector Machine med hjälp av GridSearchCV

# Resultat och Diskussion

Denna del kommer att bestå av resultat från modellerna samt diskussion om resultaten och hur man kan eventuellt nå bättre resultat.

## Resultat, Evaluering och Test

Här presenteras alla resultat från de olika testerna som gjordes för att nå slut resultatet.

### Modelval

Tabell 4.1 visar Accuracy för varje modell med sin standard hyperparametrar. Dessa modeller har tränats med tränings data och sedan blivit validerade med validerings data.

**Tabell**

|  |  |
| --- | --- |
| **Accuracy för olika modeller** | |
| Random Forest Classifier | 0.97 |
| Extra Trees Classifier | 0.97 |
| Support Vector Machine | 0.98 |
| K-Nearest Neighbors Classifier | 0.97 |
| Decision Tree Classifier | 0.87 |

Table 4.1. Modellers Accuracy

### Evaluering

Här kommer modellernas prestanda att presenteras med hjälp av Confusion Matrix, Accuracy, Precision, Recall och F1- score. Modellerna har tränats med tränings data och validerats med validerings data för att kunna mäta dessa metriker.

#### Random Forest Classifier

Tabell med utvärderingsmetriker och Confusion matrix för modellen.

**Tabell**

|  |  |  |  |
| --- | --- | --- | --- |
| **Siffror** | **Precision** | **Recall** | **F1-score** |
| 0 | 0.98 | 0.98 | 0.98 |
| 1 | 0.98 | 0.98 | 0.98 |
| 2 | 0.95 | 0.98 | 0.96 |
| 3 | 0.97 | 0.96 | 0.97 |
| 4 | 0.96 | 0.97 | 0.97 |
| 5 | 0.97 | 0.97 | 0.97 |
| 6 | 0.98 | 0.98 | 0.98 |
| 7 | 0.98 | 0.96 | 0.97 |
| 8 | 0.96 | 0.96 | 0.96 |
| 9 | 0.96 | 0.95 | 0.95 |
|  |  |  |  |
| **Medelvärde** | 0.97 | 0.97 | 0.97 |
| **Accuracy** | 0.97 |

Table 4.2 Random Forest Classifiers olika utvärderingsmetriker och dess värden för varje siffra

**Confusion Matrix**

**A graph of numbers and a graph of numbers

Description automatically generated**

Figur 4.3. Confusion Matrix för Random Forest Classifier

#### Extra Trees Classifier

Tabell med utvärderingsmetriker och Confusion matrix för modellen.

**Tabell**

|  |  |  |  |
| --- | --- | --- | --- |
| **Siffror** | **Precision** | **Recall** | **F1-score** |
| 0 | 0.98 | 0.99 | 0.98 |
| 1 | 0.98 | 0.99 | 0.99 |
| 2 | 0.96 | 0.97 | 0.97 |
| 3 | 0.98 | 0.97 | 0.97 |
| 4 | 0.97 | 0.97 | 0.97 |
| 5 | 0.98 | 0.98 | 0.98 |
| 6 | 0.99 | 0.98 | 0.98 |
| 7 | 0.97 | 0.97 | 0.97 |
| 8 | 0.97 | 0.97 | 0.97 |
| 9 | 0.97 | 0.96 | 0.96 |
|  |  |  |  |
| **Medelvärde** | 0.97 | 0.97 | 0.97 |
| **Accuracy** | 0.97 |

Table 4.4 Extra Trees Classifier olika utvärderingsmetriker och dess värden för varje siffra

**Confusion MatrixA graph of numbers and a pencil

Description automatically generated**

Figur 4.5. Confusion Matrix för Extra Trees Classifier

#### K-Nearest Neighbors

Tabell med utvärderingsmetriker och Confusion matrix för modellen.

**Tabell**

|  |  |  |  |
| --- | --- | --- | --- |
| **Siffror** | **Precision** | **Recall** | **F1-score** |
| 0 | 0.99 | 0.99 | 0.99 |
| 1 | 0.96 | 1.00 | 0.98 |
| 2 | 0.99 | 0.97 | 0.98 |
| 3 | 0.97 | 0.97 | 0.97 |
| 4 | 0.97 | 0.96 | 0.97 |
| 5 | 0.97 | 0.97 | 0.97 |
| 6 | 0.98 | 0.99 | 0.99 |
| 7 | 0.97 | 0.97 | 0.97 |
| 8 | 0.99 | 0.94 | 0.97 |
| 9 | 0.96 | 0.96 | 0.96 |
|  |  |  |  |
| **Medelvärde** | 0.97 | 0.97 | 0.97 |
| **Accuracy** | 0.97 |

Figur 4.6 K-Nearest Neighbors olika utvärderingsmetriker och dess värden för varje siffror

**Confusion Matrix**

**A graph of numbers and a graph of numbers

Description automatically generated**

Figur 4.7. Confusion Matrix för K-Nearest Neighbors

#### Support Vector Machine

Tabell med utvärderingsmetriker och Confusion matrix för modellen.

**Tabell**

|  |  |  |  |
| --- | --- | --- | --- |
| **Siffror** | **Precision** | **Recall** | **F1-score** |
| 0 | 0.99 | 0.99 | 0.99 |
| 1 | 0.99 | 1.00 | 1.00 |
| 2 | 0.97 | 0.99 | 0.98 |
| 3 | 0.99 | 0.98 | 0.99 |
| 4 | 0.98 | 0.98 | 0.98 |
| 5 | 0.98 | 0.98 | 0.98 |
| 6 | 0.99 | 0.99 | 0.99 |
| 7 | 0.99 | 0.99 | 0.99 |
| 8 | 0.98 | 0.98 | 0.98 |
| 9 | 0.98 | 0.97 | 0.98 |
|  |  |  |  |
| **Medelvärde** | 0.98 | 0.98 | 0.98 |
| **Accuracy** | 0.98 |

Table 4.8 Support Vector Machine olika utvärderingsmetriker och dess värden för varje siffra

**Confusion Matrix**

A graph of numbers and a number chart

Description automatically generated with medium confidence

Figur 4.9. Confusion Matrix för Support Vector Machine

### Test

Detta är resultatet på hur den bästa modellen från evaluerings steget. Resultatet presenteras exakt som i evaluerings steget. Skillnaden är att vi testar modellen på vår testdata för att se hur den reagerar på helt osedda data. Dessa värden kom från Support Vector Machine då den modellen prestera bäst under evaluerings steget.

**Tabell**

|  |  |  |  |
| --- | --- | --- | --- |
| **Siffror** | **Precision** | **Recall** | **F1-score** |
| 0 | 0.98 | 0.99 | 0.99 |
| 1 | 0.99 | 0.99 | 0.99 |
| 2 | 0.98 | 0.98 | 0.98 |
| 3 | 0.98 | 0.99 | 0.98 |
| 4 | 0.99 | 0.98 | 0.99 |
| 5 | 0.98 | 0.98 | 0.98 |
| 6 | 0.99 | 0.99 | 0.99 |
| 7 | 0.98 | 0.98 | 0.98 |
| 8 | 0.98 | 0.98 | 0.98 |
| 9 | 0.98 | 0.97 | 0.97 |
|  |  |  |  |
| **Medelvärde** | 0.98 | 0.98 | 0.98 |
| **Accuracy** | 0.98 |

Table 4.1.1 Support Vector Machine olika utvärderingsmetriker och dess värden för varje siffra på test data.

**Confusion Matrix**

A graph of numbers and a number

Description automatically generated

Figur 4.1.2 Confusion Matrix för Support Vector Machine utfört på testdata

### Egna Bilder

Detta är resultatet av den bästa modellens prestanda på mina egna handskrivna bilder. Resultatet kommer visa vad modellen prediktera och visa det faktiska värdet är samt hur mycket rätt den fick i procent.



Figur 4.1.3 Exempelbild på egenskriven siffra

|  |  |  |
| --- | --- | --- |
| **Faktiska Siffror** | **Predikterade siffor** | **Rätt/Fel** |
| 1 | 8 | Fel |
| 2 | 2 | Rätt |
| 3 | 3 | Rätt |
| 4 | 8 | Fel |
| 5 | 5 | Rätt |
| 6 | 5 | Fel |
| 7 | 2 | Fel |
| 8 | 8 | Rätt |
| 9 | 8 | Fel |
| 0 | 3 | Fel |
| **Antal rätt i procent** | 40% |

Table 4.1.4 Tabell med resultat på hur Support Vector Machine prediktera egna handskrivna bilder.

## Diskussion

Här kommer resultaten att diskuteras och ge insikter till varför fick dem svaren som vi fick.

### Modelval

Utifrån tabell 4.1 ser vi att många modeller har väldigt lika om inte samma Accuracy med sin standard hyperparametrar och på tränings datan. Den modell som det gick värre för var Decision Tree Classifier som inte nådde upp till de andras Accuracy. Vad detta kan bero på är att själva modellen är för simpel och behöver flera träd för att kunna prediktera bättre, som det är med till exempel Random Forest modellen. Eftersom de andra modellerna är så nära i Accuracy, fick de testas vidare och hitta försöka hitta deras bästa hyperparametrar, då det är för tidigt att bestämma vilken som är bäst ut av dem.

### Evaluering

Från tabellerna ur kapitel 4.1.2 evaluering kan vi se att modellerna är har samma Accuracy som med sina standard hyperparametrar. Detta tyder på att en marginell ändring har hänt under justeringen av hyperparametrarna. De standardvärdena hyperparametrarna blev oftast de bästa värdena för modellerna och därmed blir det inte så stor ändring i prestanda när vi validerar dessa modeller. Det enda sättet att öka prestandan med hyperparametrar och GridSearchCV är att testa flera hyperparametrar tillsamman men nackdelen är att det tar alldeles för lång tid att hitta dessa parametrar.

När vi kollar på de olika Confusion matrixes ser vi att modellerna presterar väldigt bra och gör inte många fel när de predikterar. Däremot finns det vissa siffror som modellen blandar ihop när den predikterar, till exempel blandar modellerna oftast upp sjuor för tvåor och nior för fyror. Detta kan bero på att de kan se väldigt lika ut om dem skrivs slarvigt och därför se ut som dem siffrorna när det egentligen inte är det. Dessa siffror kallas för outliers och för att förbättra modellen bör dessa outliers tas bort då de inte hjälper modellen att lära sig.

Utav alla modeller som har evaluerats kan vi konstatera att Support Vector Machine är den bästa modellen utav de urvalet av modeller. Den har högst värden utav alla utvärderingsmetriker och ser bäst ut när man jämför den med alla Confusion Matrixes. Därför väljs denna för att testas på testdata för att se så modellen inte är överanpassad och fungerar bra på osedd data.

### Test

På tabell 4.1.1 och figur 4.1.2 ser vi ungefär samma resultat som i evalueringen vilket tyder på att modellen är bra anpassad för osedd data och inte över eller underanpassad. Detta betyder att vi har hittat en modell som vi kan använda för att prediktera helt nya bilder som vi ger den och de bör ge oss samma resultat.

### Egna Bilder

Ur tabell 4.14 visar det sig att modellen inte bra på att prediktera nya handskrivna siffror då de bara får en Accuracy på 40% i stället för 98% som vi hade innan. Detta beror inte på att modellen är dålig eller överanpassad, utan att det är den data som vi ger den som är dålig. Bilderna som har använts har inte en helt slät vit bakgrund vilket leder till att det blir mycket brus i bilden när den ska skalas ner, vilket påverkar hur modellen predikterar vad som är på bilden. Ger man modellen dålig data så får man också ut dåliga prediktioner. Detta kan lösas genom att ta bättre bilder med bättre bakgrund och tydligare siffror som gör det enklare för modellen, men då uppstår problemet att det inte efterliknar verkligheten och varför ska vi då använda modellen. En annan lösning är att försöka bli av med bruset så att man bara har kvar siffrorna som man vill prediktera.

# Slutsatser

Vi kan konstatera att det finns många modeller som passar bra för bildigenkänning och att det kanske inte finns en modell som är solklar för detta MNIST-datasetet. Utifrån urvalet som vi har kan vi dock konstatera att den bästa modellen för MNIST-Datasetet är förmodligen Support Vector Machine då den gav bäst resultat genom hela projektet. Däremot finns det plats för förbättringar i metoder som kan leda till att en annan modell predikterar bättre.

När det kommer till att prediktera egna handskrivna siffror fick modellen problem, men det måste inte bero på själva modellen, utan hur bilderna är tagna och hur den är mottagen av modellen. Däremot fick vi några prediktioner rätt vilket tyder på att den kan prediktera rätt siffror men inte med samma säkerhet som med testdatan.

# Teoretiska frågor

1. Träningsdata används för att träna en modell så den kan lära sig och kunna användas för att senare kunna prediktera på annan data. Valideringsdata är till för att validera en modell efter den har tränats och kunna utveckla modellen prestanda. Validerings data kan användas vid flera tillfällen. Testdata är till för att testa modellen i slutskedet. Tanken är att man vill testa sin finjusterade modell på data som den inte har sett för att se hur modellen verkligen presterar.

2. Julia bör använda sig av korsvalidering för att hitta den bästa modellen. Med korsvalidering tränas modellerna med data som har delats upp i olika delmängder där en av delmängderna är valideringsdata. På så sätt kan hon hitta den bästa modellen.

3. Regressionsproblem tar hand om data där det finns linjära och polynoma samband. Det kan vara till exempel att man använder linjär regression för att prediktera lön baserat på ålder, kön och utbildning.

4. Root Mean Square Error kan tolkas som medelavstånded mellan vårt predikterade och det faktiska värdet. Värdet tas i kvadrat för att få värdena positiva och sedan tar man kvadratroten på värdet för att kunna tolka det med samma enhet som de predikterade och faktiska värdena.

5. Klassificerings problem tar hand om data som är uppdelat i olika klasser. Ett exempel kan vara hur en Support Vector Machine ska försöka urskilja och prediktera mellan en hund och en katt. Confusion Matrix är en tabelllayout som visar raderna som faktiska värden och columner som modellens predikterade. Tabellen ger en bra insikt på hur modellen fungerar och vart den gör rätt och fel.

6. K-means är en unsupervised clustering modell, som försöker hitta sammanhang i ostrukturerade data och gruppera dem. Det kan tillämpas inom mataffärer för att hitta samband mellan varor och köp. till exempel att tacokrydda vanligtvis köps med tortilla.

7. Dessa tre metoder används för att göra om all icke numerisk data till numerisk data. Detta görs för att modeller kan bara hantera numerisk data. Ordinal encoder fungerar genom att rangordna datan från 0 till n. Detta kan vara bra vid tillfällen då man har data som kan grupperas med en bättre och sämre plats. Det kan vara t.ex. att förstaplats i ett race som har data ’första plats’ görs om till 0 och ’andra plats’ görs om till 1 och så vidare. One-hot-encoding gör om datan till ett binärt system med 1 för sant och 0 för icke sant. T.ex. vill man göra om färger på tröjor till ett binärt system. Om man kan välja mellan röd, grön och blå, kan röd definieras som [1, 0, 0], grön [0, 1, 0] och blå [0, 0, 1]. Dummy encoding fungerar väldigt likt till one-hot-encoding men skillnaden är att man tar bort en variabel och låter det värdet med bara 0 definieras som ett värde. T.ex. kan man definiera röd som [1, 0], grön [0, 1] och blå som [0, 0]. Dummy encoding krävs vid regressions problem.

8. Julia har rätt i att datan måste tolkas först innan man gör en ordinal eller en nominal encoding. För allting kan inte rangordnas, men efter man har gjort den tolkningen så kan man säga att det antingen är det ena eller det andra, det kan inte vara både ordinalt och nominalt.

9. Streamlit är en open source data science library som kan användas till för att skapa webbapplikationer och dela sina data projekt.

# Självutvärdering

1. Utmaningar du haft under arbetet samt hur du hanterat dem.   
   Att skriva en rapport har varit utmanande. Hantera det genom att bita i det sura äpplet och fortsätta kämpa.
2. Vilket betyg du anser att du skall ha och varför.   
   VG hade inte varit fel.
3. Något du vill lyfta fram till Antonio?

Nej

# Appendix A

En streamlit webbapplikation har skapats för att göra det tillgängligt för andra att testa sina egna bilder. Den fungerar genom att man tar en bild på sin telefon, skickar in den till webapplikationen och sen får man en prediktion på vad siffran är. Webb applikationen kan ta tid att lada upp då den måste träna upp modellen, men efter första träningen så ligger det sparat i dina cookies.

A screenshot of a computer

Description automatically generated

Figur 1. Bild på webapplikation innan man laddat upp en bild

A screen shot of a number

Description automatically generated

Figur 2. Bild på webapplikation efter man laddat upp en bild med modellens prediktion.

# Källförteckning

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*. (2. uppl). O’Reilly Media

Harrison. O. (2018) Machine Learing basics with the K-Nearest Neighbors Algorithm. Hämtad 18 mars, 2024, från Towards Data Science sida https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

LeCun, Y., Cortes, C., Burges, C., (2014) The MNIST database of handwritten digits with 784 features. Hämtad mars 2024, från OpenML sida https://openml.org/search?type=data&status=active&id=554&sort=runs