The Enchanted Numerologist

Finding fun in Machine Learning



Missi Hansson

EC Utbildning

Machine Learning

202403

# Abstract

Detta projekt undersöker användningen av maskininlärningstekniker för att identifiera och klassificera handskrivna siffror med syftet att utveckla en applikation. Genom att använda olika etablerade modeller på den välkända MNIST-databasen, har projektet utforskat möjligheterna att automatisk känna igen text. Fokus har legat på att uppnå hög noggranhet. Resultaten visar att en Support Vector Machine (SVM)-modell, finjusterad med Principal Component Analysis (PCA) och MinMaxScaler, kan uppnå imponerande resultat. Slutligen diskuteras behov för vidare forskning och utveckling av bildförbehandlingstekniker, samt modellernas förmåga att hantera verkliga data.

Innehållsförteckning

[Abstract 2](#_Toc161996699)

[2 Inledning 1](#_Toc161996700)

[2.1 Syfte 1](#_Toc161996701)

[2.2 Frågeställningar 1](#_Toc161996702)

[3 Theory 2](#_Toc161996703)

[3.1 Supervised Learning 2](#_Toc161996704)

[3.2 Klassobalans 2](#_Toc161996705)

[3.3 Indelning av Data 2](#_Toc161996706)

[3.3.1 Stratified Train-Test Split 2](#_Toc161996707)

[3.3.2 Vanlig Train-Test Split 2](#_Toc161996708)

[3.4 Optimeringstekniker 3](#_Toc161996709)

[3.4.1 MinMaxScaler 3](#_Toc161996710)

[3.4.2 Dimensionalitetsreduktion med PCA 3](#_Toc161996711)

[3.5 Modeller 3](#_Toc161996712)

[3.5.1 SVM 3](#_Toc161996713)

[3.5.2 Logistic Regression 3](#_Toc161996714)

[3.5.3 KNN 4](#_Toc161996715)

[3.5.4 Random Forest 4](#_Toc161996716)

[3.5.5 Voting Classifier 4](#_Toc161996717)

[3.6 Metrics 4](#_Toc161996718)

[3.6.1 Accuracy 4](#_Toc161996719)

[3.6.2 Precision 5](#_Toc161996720)

[3.6.3 Recall 5](#_Toc161996721)

[3.6.4 F1 Score 5](#_Toc161996722)

[3.7 Förbehandling av ny data 5](#_Toc161996723)

[3.7.1 Grayscale 5](#_Toc161996724)

[3.7.2 Sharpen 5](#_Toc161996725)

[3.7.3 Threshold 6](#_Toc161996726)

[3.7.4 Morphology 6](#_Toc161996727)

[3.7.5 Etikettering och Klippning 6](#_Toc161996728)

[3.7.6 Rotation 6](#_Toc161996729)

[4 Metod 7](#_Toc161996730)

[4.1 Exploratory Data Analysis (EDA) 7](#_Toc161996731)

[4.2 Dataförberedelse 8](#_Toc161996732)

[4.3 Modeller 8](#_Toc161996733)

[4.3.1 Resultat och Validering 8](#_Toc161996734)

[4.3.2 Modellutvärdering på Testdata 9](#_Toc161996735)

[4.4 Apputveckling 9](#_Toc161996736)

[4.4.1 Bildpreprocessning 9](#_Toc161996737)

[4.4.2 Testa Appen 9](#_Toc161996738)

[5 Resultat och Diskussion 10](#_Toc161996739)

[5.1 Svar på Frågeställningar 11](#_Toc161996740)

[6 Slutsatser 12](#_Toc161996741)

[7 Teoretiska frågor 13](#_Toc161996742)

[8 Självutvärdering 15](#_Toc161996743)

[9 Referenser 16](#_Toc161996744)

# Inledning

I den digitala världen vi lever i idag blir det allt viktigare med program som kan läsa och förstå text automatiskt. Detta är inte bara något för forskare och stora företag. Nej, det är även superanvändbart i våra dagliga liv. Tänk dig att du kan få din dator att läsa och förstå dina handskrivna lappar eller att posten sorteras snabbare med hjälp av sådan teknik. Tack vare stora framsteg inom maskinlärning och datavetenskap, jobbar smarta hjärnor med att utveckla system som kan tolka handskriven text. Det här öppnar upp för en massa nya sätt att göra saker snabbare och smartare inom många olika områden.

## Syfte

Syftet med detta projekt är att utveckla en applikation som använder maskininlärningstekniker för att identifiera handskrivna siffror med hög noggrannhet. Applikationen siktar på att demonstrera potentialen och tillämpbarheten av maskininlärningsmodeller, specifikt inom området av att känna igen text i bilden. För att göra det här kommer vi använda en välkänd samling av bilder med siffror som kallas MNIST.

Användningen av avancerad teknik eller komplicerade algoritmer kommer inte att prioriteras. Fokus ligger i stället på grundläggande maskininlärningsmetoder som har bevisat sin effektivitet och tillgänglighet. Genom att utforska och experimentera med dessa etablerade tekniker, uppnås projektets mål utan onödig komplexitet. Strategin är att hålla processen så enkel som möjligt, vilket underlättar förståelsen, implementeringen och tillämpningen av utvecklade lösningar. Denna enkla metod gör projektet lättare att hantera och visar hur väl maskininlärning kan lösa riktiga problem.

För att uppfylla syftet så kommer vi besvara följande frågeställningar:

## Frågeställningar

1. Kan en enkel modell uppnå en noggrannhet på över 95% vid prediktering av MNIST-data?
2. Vilken model prestera bäst för att prediktera MNIST (Support Vector Machnie, Logistic Regression, K Nearest Neighbor, Random Forest, Voting Classifier?
3. Efter att ha testat modellen i en kontrollerad testmiljö, uppnår den samma presterande i verkliga situationer?

# Theory

I det här avsnittet några teorier gås igenom som är viktig för att bättre förstå det här projektet. Det hjälper med varför vissa beslut tas.

## Supervised Learning

Supervised Learning är en modell inom maskininlärning där modellen tränas med hjälp av data där det som sätts in har kända utvärden, som kallas etiketter. Med andra ord kan man säga att Supervised Learning är ”Väglett lärande”. (Prgomet, Introduktion till Maskininlärning, 2024)

Y = Beroende variabler / Label.

X = Oberoende variabler / Feature.

𝜀 = En slumpmässig felterm eftersom sambandet inte är deterministiskt.

𝛽\_0 & 𝛽\_1 = Parametrar.

## Klassobalans

När man tränar en maskininlärningsmodell för igenkänning av handskrivna siffror är det viktigt att ta hänsyn till hur ofta varje siffra dyker upp i datamängden. En obalans där vissa siffror är överrepresenterade kan leda till att modellen blir bättre på att känna igen just dessa siffror och sämre på de som är mer sällsynta. För att hantera denna obalans kan olika metoder användas. Det inkluderar justerar representation av klasserna i träningsdata, och specialanpassade utvärderingsmått, som tar hänsyn till obalansen vid prestandamätning. (Géron, 2019)

## Indelning av Data

### Stratified Train-Test Split

Om man är osäker på att fördelningen av klasser i ett dataset är jämn, är stratifierad uppdelning en bra metod att använda. Stratifierad uppdelning hjälper till att se till att varje klass är representerad proportionellt, både i tränings- och testsetet, vilket gör att modellerna kan jämföras rättvist. Kort förklarat, stratifierad uppdelning delar datat så att varje del har samma procentandel av varje klass som det totala datasetet. Det hjälper till att förhindra snedvridna resultat på grund av ojämn klassfördelning. (Géron, 2019)

### Vanlig Train-Test Split

En bild som visar text, skärmbild, Teckensnitt, linje

Automatiskt genererad beskrivningNär man arbetar med balanserade datamängder och snabbt vill utvärdera en modells prestanda är den vanliga uppdelningen för träning och test det självklara valet. Genom att tillämpa denna metod på alla modeller kan man se hur varje modell presterar när datat slumpmässigt delas upp. Det ger omedelbar insikt i modellernas förmåga under slumpmässiga förhållanden. Denna uppdelning är vanligtvis slumpmässig för att säkerställa en jämn fördelning av data. Vanligtvis används cirka 70-80% av data för träning och resterande 20-30% för testning. Detta tillvägagångssätt ger en snabb bedömning av modellens förmåga att generalisera till ”ny” data baserat på den information som den har lärt sig under träningen. (Prgomet, Introduktion till Maskininlärning, 2024)

Figur 3‑1 Delning av data för att träna och testa en modell.

## Optimeringstekniker

### MinMaxScaler

MinMaxScaler är en metod för att justera storleken på datans egenskaper så att de passar inom ett visst intervall, vanligtvis mellan 0 och 1. Detta görs genom att för varje värde subtrahera den minsta värdet i funktionen och sedan dela resultatet med skillnaden mellan det maximala och det minsta värdet. Genom att använda MinMaxScaler kan alla egenskaper i datan få en liknande skalning utan att förändra formen på deras distribution. Detta är särskilt användbart för algoritmer som är känsliga för storleken på oberoende data, eftersom det hjälper till att säkerställa att ingen enskild egenskap dominerar på grund av sin storlek. (Pedregosa, 2011)

### Dimensionalitetsreduktion med PCA

Dimensionalitetsreduktion med PCA (Principal Component Analysis) är en teknik som minskar antalet ”features” i datan genom att välja ut de mest betydelsefulla som förklarar största möjliga del av variansen. Det gör det genom att omvandla de ursprungliga ”features” till en ny uppsättning av ”features” ordnade så att de första få fångar så mycket av variationen i datan som möjligt. Detta kan vara särskilt användbart för att förenkla komplexa datamängder, förbättra algoritmers effektivitet och hjälpa till med visuell tolkning av datan. Genom att minska dimensionerna kan man undvika ”overfitting” och göra dataanalysen mer hanterbar. (Géron, 2019)

## Modeller

### SVM

Support Vector Machine (SVM) försöker hitta den optimala gränsen, eller hyperplanet, som bäst separerar olika klasser i ett dataset. Konceptet baseras på att hitta den gräns med största möjliga marginal, vilket innebär det största avståndet mellan de närmaste punkterna av varje klass.

Grunden för SVM är linjär separation, där data punkter kan skiljas åt med en rät linje (i två dimensioner) eller ett plan (i fler dimensioner). Men i många ”real world” problem är datan inte linjärt separerbar. Det är här SVM's kraftfulla verktyg, som ”kernel trick”, kommer in. ”Kernel trick” gör det möjligt för SVM att hantera icke-linjärt separerbara data genom att kartlägga mappa träningsdata till ett högre dimensionellt utrymme där det blir möjligt att hitta en linjär separation.

En vanlig kernel som används är Radial Basis Function (RBF), som använder avståndsmått mellan datapunkter och gör det möjligt för SVM att flexibelt anpassa sig till data genom att hitta en mjukare gräns som bättre hanterar överlappande datapunkter. (Prgomet, Introduktion till Maskininlärning, 2024)

### Logistic Regression

Logistisk regression är en statistisk metod som letar efter en samband mellan variabler och chansen för att en viss händelse ska inträffa. Det fungerar genom att använda logistiska funktionen som ger ett värde mellan 0 och 1. Detta värde representerar sannolikheten för att observationen tillhör en klass, och är baserat på en log-odds transformation av linjära förutsägelser. För att översätta detta till en sannolikhet, appliceras den logistiska funktionen på den linjära kombinationen, vilket ger följande formel:

P är sannolikheten för ”Ja”

I multiklass klassificeringsscenarion användas en multiklass strategi såsom OvR (One-versus-Rest). I OvR skapar man en separat modell för varje klass som förutsäger sannolikheten för att en observation tillhör just den klassen jämfört med alla andra klasser. (Prgomet, Linjär Regression & Logistic Regression, 2024)

### KNN

I KNN använder man närliggande träningsdata för att förutsäga klassen för ett nytt exempel. För att klassificera ett nytt exempel beräknar algoritmen avståndet till varje punkt i träningsdatan och väljer de k närmaste grannarna. Därefter används majoritetsröstning (för klassificering) av de k närmaste grannarnas etiketter eller värden för att bestämma klassen för det nya exemplet. (Géron, 2019)

### Random Forest

Random Forest kombinerar flera beslutsträd för att förbättra modellens noggrannhet och minska risken för ”overfitting”. Varje träd i "skogen" tränas på en slumpmässigt vald undergrupp av datamängden med återläggning, en process känd som bootstrap sampling. När det är dags för klassificering tar Random Forest ett majoritetsbeslut från alla träd för att förutsäga den slutliga klassen. Varje träd ger en röst på klassen och den klass som får flest röster blir den förutsagda klass för den nya datapunkten. (Pedregosa, 2011)

### Voting Classifier

En Voting Classifier kombineras fler olika modeller för att göra en gemensam förutsägelse. I stället för att lita på en enskild modells förutsägelse, använder en Voting Classifier ett "röstningssystem" där varje enskild modell ”röstar" för sin förutsagda klass, och klassen som får flest röster blir den slutliga förutsägelsen.

Det finns två huvudsakliga typer av röstning:

* Hard Voting: Varje modell i ensemblen får rösta på en klass, och klassen som får flest röster väljs som slutresultat. Här räknas endast rösterna, inte sannolikheten.
* Soft Voting: Modellerna uppskatta sannolikheter för klasserna, så räknas medelvärdet av dessa sannolikheter, och klassen med högst genomsnittlig sannolikhet väljs som slutresultat. Gäller modeller som kan rösta. (Géron, 2019)

## Metrics

För att bedöma maskininlärningsmodellers prestanda används olika mätvärden. Dessa mätvärden ger information om hur väl modellerna klarar specifika delar av förutsägningsuppgiften. (Prgomet, Klassificering, 2024)

### Accuracy

Accuracy är en vanlig mätningsmetod som visar hur många exempel som klassificerats korrekt jämfört med det totala antalet exempel. Det är enkelt att förstå och beräkna, men det kan vara vilseledande när datamängden är obalanserad, som när en klass är betydligt vanligare än andra.

### Precision

Precision mäter andelen korrekta positiva förutsägelser i förhållande till det totala antalet positiva förutsägelser som gjorts av modellen.

Notera att precision är extra viktig är avgörande när kostnaden för en falsk positiv är hög.

### Recall

Recall mäter hur bra modellen identifierar alla relevanta fall inom en given kategori.

Notera att recall är extra viktig i tillämpningar där kostnaden för att missa ett positiv fall är hög.

### F1 Score

F1 Score är ett mått som kombinerar precision och recall i en enda metrik genom att ta deras genomsnittliga medelvärde. Det ger en balans mellan precision och recall och är användbart när det behövs en balans mellan att identifiera alla relevanta exempel (recall) och att vara korrekt när en positiv förutsägelse görs (precision).

Notera att F1 Score är särskilt användbart i situationer där det finns en obalans i klassfördelningen.

## Förbehandling av ny data

Innan specifika bildbehandlingsmetoder är det väsentligt att förstå vikten av förbehandling av bilddata. Detta inkluderar att omvandla färgbilder till gråskala för att förenkla analysen, skärpa bilden för att framhäva viktiga drag och tillämpa tröskelvärdesinställningar för att skilja objekt från bakgrunden, med mera. Dessa steg är kritiska för att förbättra bildbehandlingsflöden och förbättra klassificeringsalgoritmernas noggrannhet. Dessa processer och metoder är grundligt beskrivna och implementerade i biblioteket scikit-image, som erbjuder ett omfattande utbud av verktyg för bildmanipulering och analys som är väl anpassade till dessa uppgifter. (Stéfan van der Walt, 2023)

### Grayscale

Konverteringen av en bild till gråskala är ett grundläggande steg i bildbehandling som reducerar komplexiteten genom att eliminera färginformation, vilket lämnar intensitetsinformationen av ljus. Bildens färginformation (oftast RGB) kombineras och reduceras till en enda intensitetskanal.

### Sharpen

”Sharpen” förbättrar bilden genom att betona gränser och detaljer. Detta gör det lättare att identifiera egenskaper och ytterlinjer i bilden, vilket kan vara särskilt användbart för att förbättra prestandan hos en klassificeringsalgoritm.

### Threshold

Otsu-metoden automatiserar processen att välja ett ”threshold” för att konvertera en gråskalabild till en binärbild. Pixelvärden över detta ”threshold” sätts till vitt och de under till svart, vilket resulterar i en binärbild. Den gör detta genom att minimera variationen inom klasserna av svarta och vita pixlar, vilket förbättrar separationen mellan objekt (förgrund) och bakgrund.

### Morphology

”Morphology” används för att rensa upp bilden från brus och små objekt samt att fylla i hål och gap i de identifierade objekt. Detta bidrar till att skapa en mer sammanhängande representation av varje objekt.

### Etikettering och Klippning

Genom etikettering av sammanhängande komponenter isoleras olika objekt i bilden. Sedan klippa bilden till den största komponentens i en ”bindingbox” för att fokusera på den specifika objekt, vilket tar bort onödig bakgrund.

### Rotation

Bilden justeras genom att rotera den enligt den uppskattade inriktningen för att standardisera dess riktning.

# Metod

Detta kapitel dyker ned i processen av att utforska MNIST-datasetet och dess handskrivna siffror, till att hitta en modell som fungerar i verkliga situationer. Resan börjar med att undersökning av MNIST dataset för att verkligen förstå de handskrivna siffrorna. Sedan följer en grundlig utforskning av data, där de viktigaste bitarna plockas ut. Efter det testas lämpliga modeller, finjusteras för att få fram de bästa resultaten och till sist avslöjas den utvalda modellen i detalj. Målet är inte bara att hitta ett sätt att känna igen siffrorna på. Det handlar också om att klargöra varför och hur dessa specifika metoder sticker ut som de bästa.

## Exploratory Data Analysis (EDA)

I den här delen utforskas det välkända MNIST-datasetet, skapat av Yann LeCun, Corinna Cortes och Christopher J.C. Burges. Datasetet, som finns tillgängligt på MNIST:s webbplats, består av 70 000 bilder på handskrivna siffror. Dessa siffror har normaliserats till en enhetlig storlek och centrerats i en bild med fast storlek på 28x28 pixlar. Bilderna har gjorts mjukare genom anti-aliasing, en teknik som minskar skarpheten vid pixelkanterna. Varje bilds 28x28-pixelrutnät motsvarar 784 unika egenskaper, där varje pixels intensitet ger värdefull information för mönsterigenkänning. Så noggrann förberedelse garanterar att varje siffra visas på ett sätt som underlättar för datorprogram att känna igen mönster och lära sig av dem.

En bild som visar typografi, Teckensnitt, svart och vit, vit

Automatiskt genererad beskrivningMålet med EDA är att förstå MNIST-datasetet genom att fokusera på fördelningen av sifferklasser och relationerna mellan pixelfunktioner. Med tanke på det väl dokumenterade renhetsgraden av MNIST-datasetet, ignoreras sökandet efter saknade värden eller avvikande observationer. För att förstå MNIST-datasetet används några tekniker:

* Utforska sifferbilder: Dyk ner i datasetet genom att titta på själva siffrorna.
  + Visa ett urval av bilder för varje siffra för att uppskatta variationen av handstilar och nyanser med hjälp av "matplotlib".
  + Generera en genomsnitts bild för varje sifferklass. Det är en viktig del av förbehandlingen av bilderna. Det fungerar som ett användbart utforskande verktyg för att bättre förstå varje sifferklass och kan vara avgörande för att upptäcka mönster och variationer i datan, även om datasetet är rent. Verktyg som används är "numpy" och "matplotlib".

Figur 4‑1 Slumpmässigt urval av varje siffra.

En bild som visar typografi, Teckensnitt, vår, spiralfjäder

Automatiskt genererad beskrivningUtforskningen avslöjar att det finns en mångfald av handstilar som representeras. Varje siffra, från 0 till 9, visar variationer i drag, form och uppbyggnad. Denna variation lyfter fram den utmaning som maskininlärningsmodeller står inför när de ska urskilja de små skillnaderna som gör en siffra unik från en annan. När man betraktar de genomsnittliga bilderna för varje siffra ser man en suddighet. Detta är inte bara en effekt av bearbetningen. Dessa genomsnittsbilder fångar hur siffrorna generellt ser ut, genom att sammanfoga unika detaljer från varje exempel till en enda bild. Denna blandning hjälper till att avslöja de mest gemensamma dragen som delas bland de varierade exemplen av varje nummer.

Figur 4‑2 Genomsnittliga bilder för varje siffra.

* En bild som visar text, skärmbild, Färggrann, Graf

  Automatiskt genererad beskrivningAnalys av Klassfördelning: Undersök hur många exempel det finns för varje siffra för att säkerställa att datasetet är balanserat.

Figur 4‑3 Countplot av klasserna

* + Användning av seaborn för att visualisera antalet exempel per sifferklass hjälper till att identifiera eventuella obalanser. Detta steg är avgörande för att förhindra eventuell modellbias mot siffror som förekommer oftare.

Grafen över klassfördelningen visar att det finns ett relativt balanserat antal exempel över sifferklasserna i datasetet. Trots att en del siffror dyker upp lite oftare eller mer sällan än andra, är skillnaderna inte stora nog att skapa en betydande snedvridning. Sådan balans är gynnsam för att träna maskininlärningsmodeller eftersom det bidrar till att ingen specifik siffra blir för dominerande eller underrepresenterad, vilket kan öka modellens precision över olika kategorier. Även om fördelningen ser jämn ut vid en första anblick, kan det fortfarande vara värt att utforska ytterligare för att säkerställa att det inte finns några subtila ojämlikheter som kan påverka modellens resultat, något som kan bli mer uppenbart när data stratifieras.

* En bild som visar skärmbild, Digitalur, klocka, design

  Automatiskt genererad beskrivningAnalys av Pixelintensitet: Detta innebär att man undersöker hur ljusa eller mörka pixlarna är över alla bilder för att avgöra om det är nödvändigt att justera dem för bättre modellprestanda.
  + För att utforska variationer i pixelintensitet använder vi "seaborn" för att skapa en ”heatmap” av genomsnittsbilderna för varje sifferklass. Genom att beräkna medelpixelvärdena för varje klass och presentera dessa som ”heatmap” kan vi visuellt identifiera områden med betydande variation.

Vid granskning av ”heatmap” är det tydligt att vissa regioner inom varje siffra representation visar konsekvent högre eller lägre intensitetsnivåer. Dessa mönster tyder på att modeller skulle kunna uppnå högre noggrannhet genom att fokusera på dessa nyckel ”pixlar”, möjligtvis genom att reducera dimensionen av datan.

Figur 4‑4 Heatmap av samtliga siffror, understryker ivariation i pixlarna.

## Dataförberedelse

Dataförberedelsen delades in i två huvudsteg för alla modeller: användning av Principal Component Analysis (PCA) för att reducera dimensioner och framhäva viktiga drag, samt tillämpning av MinMaxScaler för att normalisera datan och säkerställa jämn skalning. Denna kombinerade approach förbättrar modellernas förmåga att göra noggranna förutsägelser.

## Modeller

För modellerna SVM, Logistisk Regression och KNN användes både PCA för dimensionell reduktion och MinMaxScaler för datanormalisering. Random Forest tillämpade enbart PCA, medan Voting Classifier kombinerade både PCA och MinMaxScaler.

### Resultat och Validering

Varje modell tränas på träningsdata och testas på valideringsdata. Resultaten sammanställt i en dataframe för att enklare jämföra deras prestationer. Metric som användas för bedömning är accuracy, men andra mått har också kollats på såsom återkallning (recall), precision och F1-score. Baserat på dessa resultat blev SVM med rbf kernel den mest lämpliga modellen för denna applikation.

### Modellutvärdering på Testdata

Efter att ha valt SVM baserat på valideringsresultaten, testades modellen på ett ”test” data. Detta för att se hur väl modellen presterar på helt ”ny” data. Samma mätvärden (accuracy, recall, percision och F1 score) används för att få en komplett bild av modellens kraft. Resultaten från testsetet bekräftar modellens förmåga att generalisera och dess prestanda i ”nya” situationer. Denna slutliga utvärdering ger ett grönt ljus för att implementera modellen i applikationen.

## Apputveckling

Nu är modellen klar för användning måste själva applikationen byggas. Följande implementerats för att behandla och klassificera användarens handskrivna siffror.

### Bildpreprocessning

Scikit image bibliotek används för applicera följande steg.

* Konvertera till gråskala
* Skärpa bilden
* Tillämpa threshold\_otsu
* Invertera bilden
* Morphology operationer:
  + binary\_opening
  + binary\_closing
  + binary\_erosion
  + binary\_dilation
* Identifiera sammanhängande delar med label
* Identifiera och beskära siffran
* Roterar siffran
* Ändra storlek på siffran och bibehåller proportionerna.
* Lägger till vit marginal runt den skalenligt anpassade siffran för att centrera den inom en 28x28 pixelsbild. Därefter plattas bilden ut till en enkel rad.
* Omformar den plattade bilden till en 2D-array med en rad och n kolumner.

### Testa Appen

Sist testas applikationen och justera bildpreprocessningsstegen tills önskat resultat uppnås. Efter detta utvärderas modellen för verklig användning genom att testa den med olika typer av bilder.

# Resultat och Diskussion

Support Vector Machine (SVM)-modellen har utmärkt sig att kunna känna igen handskrivna siffror, med en imponerande noggrannhet på 98,79 %, vilket starkt skiljer sig från Logistic Regression som noterade en noggrannhet på 92,31 %. Användningen av Principal Component Analysis (PCA) har varit avgörande i detta sammanhang, eftersom det inte bara reducerar data till dess mest betydelsefulla dimensioner utan också förstärker viktiga drag som gör att modeller som SVM kan prestera ännu bättre. Genom att fokusera på de mest informativa egenskaperna i datan, kan PCA minska ”overfitting” och förbättra beräkningshastighet.

Det är intressant att notera att även om Voting Classifier kombinerar flera modellers prediktioner, vilket teoretiskt skulle kunna leda till bättre prestanda, så var det inte i detta projekt då SVM presterade bättre. Detta väcker frågor om hur andra ensemblemetoder såsom bagging eller boosting kanske skulle hantera svagheterna hos de individuella modellerna mer effektivt. Desa metoder bygger på principen om att skapa starkare, mer robusta modeller genom att integrera resultat från flera olika modeller som tillsammans kompenserar för varandras brister.

När det gäller datauppdelningen konstaterades att även med MNIST, ett dataset som anses balanserat, förbättrades modellernas ”accuracy” med Stratified Shuffle Split. Det visar vikten av att ha balans bland klassarna, som till exempel med MNIST fanns det lite färre bilder av femman. Denna anpassning av träningsdata förbättrade alla modellers ”accuracy”, med undantag för Voting Classifier.

En bild som visar text, skärmbild, Teckensnitt, nummer

Automatiskt genererad beskrivning

Figur 5‑1 Resultat på dem olika modeler.

Vidare observerades att när SVM modellen implementeras i praktiska applikationer, sänktes prestationen. Detta bero på skillnader i datakvalitet och variationer som inte helt fångas upp av träningsdatasetet. Exempelvis kan siffran åtta i MNIST vara perfekt jämfört med verkliga handskrivna åttor i verklighets test variera i stil, orientering och tjocklek. Dessa skillnader kan resultera i att en modell som tränats på MNIST inte presterar lika bra på verklig data eftersom den kanske inte kan hantera dessa variationer.

En utmaning framåt ligger i att utveckla modeller som inte bara presterar väl på balanserade och förbehandlade dataset som MNIST, utan som även är kapabla att korrekt identifiera och klassificera handskrivna siffror med stor variation i stil, storlek och orientering. En underskrift på ett kvitto kan vara sirlig och tydlig, medan en hastigt nedklottrad anteckning kan vara nästan oläslig. Därför måste förbehandlingsteknikerna anpassas för att hantera denna variation. Förutom att normalisera bakgrunden till att vara uniformt vit, kan tekniker som normalisering av ljusstyrka och kantdetektering användas för att förbättra modellernas förmåga att urskilja siffror från brus. Att förstå och minska gapet mellan modellprestation i testmiljö och den faktiska användningen är centralt för att vidareutveckla teknologin.

## Svar på Frågeställningar

1. Kan en enkel modell uppnå en noggrannhet på över 95% vid prediktering av MNIST-data?

Ja, i detta test nådde SVM-modellen en accuracy över 95%. Det bästa modell var nästan 99%. Det visar att även enkla modeller kan vara riktigt bra om man ställer in dem rätt och tar hand om datan innan man använder den.

1. Vilken model prestera bäst för att prediktera MNIST (Support Vector Machnie, Logistic Regression, K Nearest Neighbor, Random Forest, Voting Classifier?

Bland alla modeller testades var det SVM-modellen med rbf kernel som klarade sig bäst. Det här bekräftar att rbf-kärnan är toppen för att hantera icke-linjär dataset.

1. Efter att ha testat modellen i en kontrollerad testmiljö, uppnår den samma presterande i verkliga situationer?

Det är inte lika rakt fram att säga hur modellen presterade med riktiga bilder jämfört med de från MNIST-databasen. Prestandan sjönk lite när verkliga bilder testades, vilket delvis kan bero på att det inte fanns så många testbilder. Dessutom var processen för att göra bilderna renare och mer lämpade för testningen ganska komplicerad. Det visar att mer jobb att förbereda bilder från den riktiga världen behövs, så att modellen kan hantera dem bättre.

# Slutsatser

Projektets mål att utveckla en applikation för identifiering av handskrivna siffror med hjälp av maskininlärningstekniker har uppnåtts, med Support Vector Machine (SVM) som den mest framgångsrik modell. Genom att använda Principal Component Analysis (PCA) och MinMaxScaler för datanormalisering, uppnådde SVM-modellen en imponerande ”accuracy”. Trots framgångarna med denna initiala studie finns det flera vägar för framtida forskning och utveckling som kan förbättra och utöka applikationens funktionalitet och tillförlitlighet.

Att fördjupa analysen och finjusteringen av de individuella modellerna är ett kritiskt nästa steg. Genom att skräddarsy varje modells hyperparametrar efter datan kan deras unika styrkor nyttja för att betydligt förbättra ”accuracy”.

Ett annat område som kräver arbete är bildförbehandlingen. Framåt måste fokus läggas på att utveckla och finjustera förbehandlingsprocesser för bilder. Detta innebär att använda verkliga data från mångfaldiga källor och situationer, inklusive handskrivna siffror från vardagliga dokument till brev. Att anpassa bildförbehandlingsteknikerna för att bättre hantera dessa variationer är avgörande för att förbättra modellens förmåga att korrekt tolka och klassificera handskrivna siffror.

Genom fortsatt utforskning och förbättring av både modellprestanda och bildförbehandlingsmetoder kommer maskininlärningstekniken att göras mer tillgänglig och användarvänlig. Detta kommer att underlätta digitalisering och automatisering på nya innovativa sätt och även säkerställas att tekniken kan tillämpas i verkliga situationer. Detta kopplas till vår nutida digitala värld där behovet av intelligenta system som kan tolka handskriven text blir allt mer betydande, inte bara inom forskningsvärlden och stora företag utan också i vårt dagliga liv. Tanken om att datorer enkelt kan förstå handskrivna anteckningar eller att posthanteringen kan bli snabbare med hjälp av dessa teknologier visar på de många nya möjligheter som öppnas för att göra våra liv mer effektiva, smartare och bättre kopplade.

# Teoretiska frågor

Besvara nedanstående teoretiska frågor koncist.

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

Träning: Används för att lära modellen att känna igen mönster.

Validering: Används för att finjustera modellens parametrar och förhindra ”overfitting” och ”underfitting”.

Test: Används för att slutligen utvärdera modellens prestanda på ”ny” data.

1. 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 ”validerings-dataset”?

Hon kan använda cross-validation för att jämföra modellernas prestanda på träningssättet, vilket ger en uppskattning av hur väl de kommer att prestera på ”ny” data. Cross-validation är en metod som delar upp träningsdatan i flera mindre delar och då används en del att träna och en annan att utvärdera.

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

Regressionsproblem handlar om att förutsäga ett kontinuerligt värde. (till exempel lön eller huspris, längd)

Exempel på modeller som kan användas är Linear Regression, Lasso, Ridge, Elastic är några exempel på modeller som kan användas på regresssionsproblem.

Tillämpningsområden kan inkludera analysera samband mellan utbildning, ålder och lön.

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

En bild som visar Teckensnitt, text, linje, diagram

Automatiskt genererad beskrivning

RMSE är ett mått på modellens prediktionsfel. Det är skillnaderna mellan faktiska och förutsagda värden. Det används för att kvantifiera hur nära modellens förutsägelser ligger till de faktiska värdena.

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

Klassificieringsproblem handlar om att förutsäga tillhörighet till en eller flera kategorier. Y variabel med andra ord är ett bestämt värde.

Exempel på modeller inkluderar Logistisk regression, Beslutsträd, K Nearest Neighbor. Tillämpningar inkluderar: spam filter, Iris dataset, MNIST dataset

En Confusion Matrix visar antalet korrekta och felaktiga förutsägelser, uppdelat per kategori; med andra ord True positive, false positive, recall (andel förutsagd positiv av faktiskt positiva), precission (andel rätt förutsags av samtliga).

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

K Means är en metod som grupperar data i K (definerat antal) antal kluster baserat på likheter. Slumpmässigt placeras klustercentra i datarummet. Därefter justerar den gradvis dessa centra för att minimera avståndet mellan datapunkter och det närmaste klustercentret. Tillämpningar inkluderar: E-handelsföretag segmentering är att segmentera sina kunder baserat på köpbeteende för att senare rikta sina marknadsföringskampanjer mer effektivt.

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

Ordinal encoding: Ordinal encoding används när det finns en ordning eller rangordning bland kategorierna i en kategorisk variabel. Exempel: Kategorierna [Högstadium, Gymnasium, Universitet] skulle bli [1, 2, 3]

One-hot encoding: One-hot encoding används när kategorierna inte har någon ordning. Varje kategori skapar en ny binär variabel, och endast en av dessa variabler är 1 (på) medan resten är 0 (av). Exempel: [Grön], [Blå], [Röd] skulle se ut som [1, 0, 0], [0, 1, 0], [0, 0, 1]

Dummy variable: Typ samma som one-hot encoding men en av kateforierna tas bort. Exempel: [Man], [Kvinna] skulle bli [1] för Man och [0] till Kvinna.

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?

Både har rätt för det beror på sammanhanget.

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?

Streamlit förvandlar kod till webbapplikationer som kan delas på några minuter.

# Självutvärdering

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

Den största utmaningen var förbehandlingsprocessen av bilderna. Jag tog mycket tid att läsa och testa olika förbehandlingstekniker. DataCamp och Scikit Image dokumentationen var min räddning. Jag hade även problem med att begränsa mig och var tvungen att slutföra arbetet på grund av tidsbegränsningen. Det speglar verkligheten och jag insåg att man inte alltid har hur mycket tid som helst för att uppnå perfektion. Begreppet "lagom" från svenskan är mycket relevant i detta sammanhang.

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

Jag har jobbat för ett VG. Jag har lagt ner mycket tid och ansträngning, strävat efter att vara grundlig och kreativ. Alltid försökt hjälpa mina klasskamrater när jag haft möjlighet. Jag känner att jag har förstått de grundläggande principerna i maskininlärning, framgångsrikt implementerat modeller i en verklig situation. Jag har stort engagemang.

1. Något du vill lyfta fram till Antonio?

Jag älskade denna kurs. Detta är definitivt något som jag vill jobba med i framtiden. Det jag verkligen uppskattar var hur vi uppmuntrades att ta de teoretiska idéerna och använda dem i verkliga situationer. Det har verkligen hjälpt mig att inte bara förstå maskininlärning utan även inse att jag själv har förmågan att tillämpa min teoretiska kunskap.

# Referenser

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

Pedregosa, F. a. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 2825-2830.

Prgomet, A. (2024, Februari). *Introduktion till Maskininlärning.* Retrieved from github.com/AntonioPrgomet/machine\_learning\_ds23: https://github.com/AntonioPrgomet/machine\_learning\_ds23/blob/main/l1/01\_introduktion\_till\_maskininl%C3%A4rning.pptx

Prgomet, A. (2024, Februari). *Klassificering.* Retrieved from Github.com/AntonioPrgomet/machine\_learning\_ds23: https://github.com/AntonioPrgomet/machine\_learning\_ds23/blob/main/l2/02\_klassificering.pptx

Prgomet, A. (2024, Februari). *Linjär Regression & Logistic Regression.* Retrieved from github.com/AntonioPrgomet/machine\_learning\_ds23: https://github.com/AntonioPrgomet/machine\_learning\_ds23/blob/main/l2/03\_linreg\_logreg.pptx

Prgomet, A. (2024, Februari). *Support Vector Machines (SVM).* Retrieved from github.com/AntonioPrgomet/machine\_learning\_ds23: https://github.com/AntonioPrgomet/machine\_learning\_ds23/blob/main/l5/07\_svm.pptx

Stéfan van der Walt, J. L.-I.-i.-i. (2023, 10 03). *Image processing in Python*. Retrieved from scikit-image: https://doi.org/10.7717/peerj.453