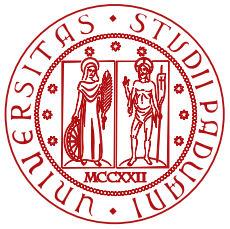

Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Triennale in Ingegneria Informatica

Classificazione delle impronte digitali mediante SVM

Laureando

Giacomo Camata

Matricola 1217996

Relatore

Ch.mo Prof. Luca Schenato

Università degli studi di Padova

Anno Accademico
2022/2023

Abstract

Oggi, le impronte digitali hanno assunto un ruolo di fondamentale importanza grazie alla loro diffusione nei dispositivi mobili. Il principale utilizzo dei sistemi di autenticazione biometrica è l'identificazione e il conseguente accesso sicuro alle informazioni. Pertanto, si è reso necessario il miglioramento della classificazione delle impronte che rappresenta un obiettivo cruciale per garantire l'efficienza e l'affidabilità di tali sistemi.

Le impronte digitali venivano utilizzate come metodo di identificazione già nel passato, ma solo l'avvento delle moderne tecnologie ha reso possibile digitalizzarle e analizzarle in modo efficiente. Per migliorare l'accuratezza dell'identificazione attraverso le impronte digitali si è reso necessario catalogarle e uno dei metodi utilizzati sono le SVM (*Support Vector Machines*), un potente strumento per la categorizzazione di dati complessi.

Quest'analisi inizia ripercorrendo la storia delle impronte digitali e le scoperte scientifiche che hanno portato al loro utilizzo come metodo di riconoscimento. Successivamente, vengono evidenziate le tecniche di elaborazione delle immagini ottenute scansionando le impronte. Infine, dopo aver introdotto la teoria degli SVM, viene presentato un semplice classificatore che ne sfrutti le potenzialità.

Indice

1 Le impronte digitali

1.1 Cenni storici

1.1.1 Primi utilizzi

In un sito archeologico della Cina nordoccidentale è stata scoperta una terracotta risalente a 6000 anni fa sulla quale sono chiaramente distinguibili le impronte digitali più antiche mai rinvenute. Tuttavia, non è noto se siano state depositate per caso o con il fine di creare motivi decorativi.

La cultura cinese è stata la prima ad utilizzare le impronte digitali come mezzo d'identificazione e ne è testimonianza un documento risalente alla dinastia Qin (221-206 a.C.) che evidenzia l'uso delle impronte delle mani come prova di autenticità. I ritrovamenti, composti da fogli di bambù, venivano arrotolati con lacci di corda e sigillati con dell'argilla. Su un lato del sigillo veniva impresso il nome dell'autore e sull'altro la sua impronta digitale con lo scopo di dimostrare la paternità del documento e impedirne la manomissione. Quindi, la validità del documento veniva confermata sia dall'impronta digitale che dal nome dell'autore. L'identificazione sia tramite nome che mediante impronta digitale divenne sempre più diffuso a seguito dell'invenzione della carta (105 d.C.), estendendosi anche ai paesi con cui la Cina intratteneva relazioni commerciali, in particolar modo Giappone e India [14].

1.1.2 Primi studi

Verso la fine del XVII secolo, con le prime pubblicazioni degli scienziati europei sulle impronte digitali, nacque la *dattiloscopia*, ovvero la scienza che studia le creste cutanee presenti sui polpastrelli delle dita.

In Europa, le impronte digitali furono descritte per la prima volta in maniera dettagliata dal dottor Nehemiah Grew nell'articolo *The description and use of the pores in the skin of the bands and feet* [7] del 1684.

Nel 1665, il fisiologo italiano Marcello Malpighi, il primo ad utilizzare il microscopio per studi medici, pubblicò *De externo tactus organo anatomica observatio* [15] in cui analizzava la funzione, la forma e la struttura della pelle.

Sebbene le impronte digitali fossero oggetto di studio già da diversi anni, la loro unicità venne riconosciuta solo nel 1788 dal medico e anatomista tedesco Johann Christoph Andreas Mayer con una pubblicazione che includeva anche dei disegni specifici [18].

Il professor Johannes Evangelista Purkinje, nel suo saggio *Beobachtungen und Versuche zur Psychologie der Sinne* [21] del 1823, classificò le impronte digitali dividendole in nove categorie, dando a ciascuna di esse un nome. Pur non avendone colto le potenzialità nel riconoscimento delle persone, il contributo di Purkinje fu significativo poiché i modelli da lui individuati (Figura 1.1) fornirono le basi per la modalità di categorizzazione moderna (anche nota come sistema di classificazione di Henry) [14].

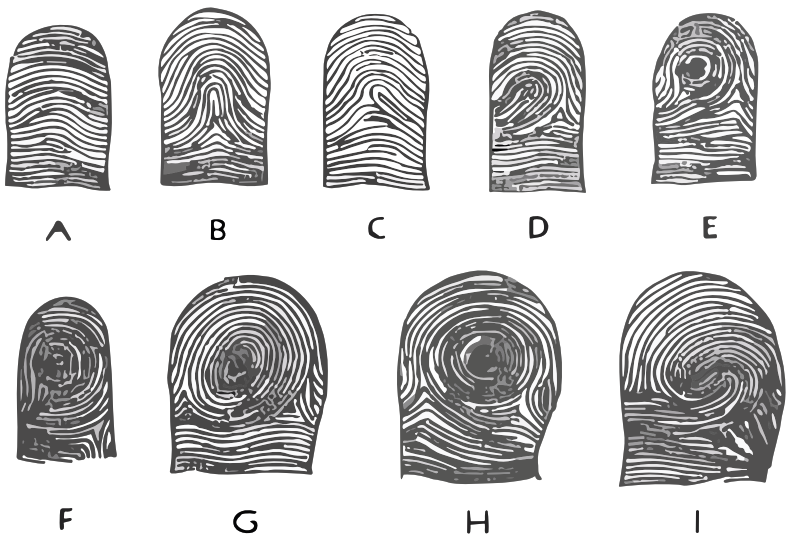

Figura 1.1: Le nove categorie individuate da Purkinje

1.1.3 Prime applicazioni

Nel 1858, Sir William James Herschel, un amministratore britannico della Compagnia delle Indie Orientali, ebbe l'idea di utilizzare l'impronta della mano destra, in sostituzione della firma, per sottoscrivere un contratto con un fornitore. Il documento venne ritenuto valido e rappresentò il primo uso delle impronte digitali su un atto ufficiale. Nel 1877, Herschel venne nominato magistrato nella regione del Bengala ed essendo il responsabile dei tribunali penali, delle prigioni e della registrazione degli atti poté istituire l'uso delle impronte digitali come metodo di riconoscimento su larga scala.

Il medico inglese Henry Faulds iniziò ad interessarsi al tema dopo aver visto delle impronte digitali impresse su dei manufatti in ceramica rinvenuti in una spiaggia giapponese. Faulds aprì un ospedale a Tsukiji, in Giappone, in cui lavorò dal 1873 al 1885 conducendo ricerche indipendenti sulla raccolta di impronte di scimmie e di esseri umani. Nel febbraio 1880, scrisse una lettera al celebre naturalista Charles Darwin, affermando che le impronte digitali sono uniche, classificabili e permanenti. Nel mese di ottobre dello stesso anno, inviò un articolo alla rivista *Nature* [5] in cui ne proponeva l'utilizzo per facilitare le indagini di polizia.

Nel 1879, Alphonse Bertillon, un impiegato della prefettura di polizia di Parigi, sviluppò l'*antropometria*, ovvero la scienza che si basa sullo studio delle misure del corpo umano per identificare le persone. Nel 1914, dopo un periodo di sperimentazione, le 10 impronte digitali furono aggiunte alle già presenti registrazioni antropometriche che comprendevano 11 misure corporee e 2 fotografiche (frontale e laterale destra).

Nel 1883, Arthur Kollmann, un medico e ricercatore di Amburgo, contribuì alla ricerca sulle impronte digitali con una pubblicazione sullo sviluppo embriologico. Nel suo libro *Der Tastapparat der Hand der menschlichen Rassen und der Affen in seiner Entwickelung und Gliederung* [12] Kollmann dimostrò che si sviluppano nel feto a partire dal quarto mese di gravidanza e sono completamente formate entro il sesto mese.

Le impronte digitali furono oggetto di interesse anche da parte di un importante scienziato inglese dell'epoca, Sir Francis Galton, cugino di Charles Darwin. Nel 1892, Galton scrisse il primo libro a riguardo intitolato *Fingerprints* [6] che ne stabiliva l'unicità, la persistenza e l'assenza di correlazione con la storia genetica di un individuo.

Galton fu il primo a definire i cosiddetti *dettagli di Galton* (Figura: 1.2) ovvero le minuzie delle impronte digitali: *bifurcation* (a,b), *lake* (c), *ridge ending* (d,e) e *point* (f).

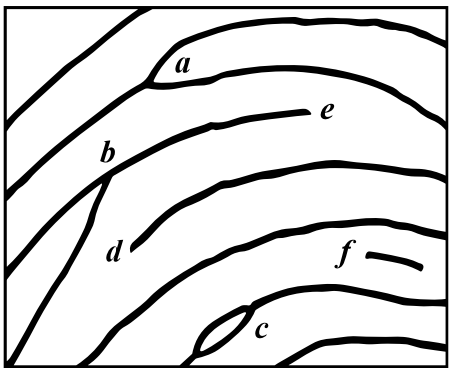

Figura 1.2: Le minuzie individuate da Galton

Un altro importante ricercatore del periodo fu Juan Vucetich, statista presso il dipartimento di polizia di La Plata, in Argentina. Nel 1891 Vucetich, dopo aver studiato le ricerche di Galton, creò un sistema di classificazione per poter schedare i criminali catturati dal suo dipartimento. Nel 1892, a Buenos Aires, un omicidio fu risolto grazie al sistema di Vucetich rendendo l'Argentina il primo paese ad affidarsi a questa nuova tecnologia per le indagini.

Nel 1894, Sir Edward Richard Henry, ispettore generale di polizia delle province basse del Bengala, studiò attentamente le peculiarità delle impronte digitali e scoprì che ognuna aveva una struttura unica, con creste e solchi che formavano modelli distinti. In seguito, collaborò con Galton per creare un sistema di classificazione affidabile chiamato *metodo di classificazione di Henry*.

Nel dicembre del 1900, il comitato di Belper decise di adottare le impronte digitali come mezzo esclusivo di identificazione dei detenuti nei paesi dell'impero britannico, dando il via all'utilizzo del sistema di classificazione di Henry su larga scala [14].

Con la rapida espansione di questo innovativo sistema di riconoscimento, gli archivi contenenti le impronte digitali divennero molto vasti, rendendo complessa l'identificazione manuale delle impronte stesse.

A partire dagli anni '60, l'FBI, il ministero degli interni nel Regno Unito e il dipartimento di polizia di Parigi iniziarono a investire nello sviluppo di sistemi automatizzati di identificazione delle impronte digitali. Nacque così l'*Automated Fingerprint Identification System* (AFIS), un sistema automatico incaricato di acquisire le impronte, estrarne le caratteristiche e verificarne la corrispondenza.

Negli ultimi anni questa tecnologia è cresciuta rapidamente anche al di fuori dell'ambito forense, fino ad essere utilizzata anche in campo civile e commerciale. Infatti, i sistemi biometrici basati sulle impronte digitali hanno acquisito una popolarità tale da diventare i più diffusi sul mercato.

1.2 Anatomia

Le impronte digitali iniziano a formarsi durante il quarto mese di gestazione, quando l'epidermide del feto cresce e inizia a ripiegarsi su se stessa creando delle zone più spesse e sporgenti, le *creste*, e altre più sottili e incavate, chiamate *valli*.

Al termine del settimo mese, il processo di formazione è concluso e il feto è dotato di impronte digitali interamente definite che rimarranno tali per tutta la sua vita.

La struttura e la posizione delle creste e delle valli è condizionata da svariati fattori ambientali: la posizione del feto, il flusso del liquido amniotico e l'ambiente uterino. Queste variabili determinano l'unicità della posizione di creste e valli, impedendo l'esistenza di due impronte completamente uguali.

Anche il patrimonio genetico dell'individuo influenza, seppur in parte, il processo di formazione delle impronte digitali. La massima somiglianza è osservabile nei gemelli monozigoti che sono gli individui che tra loro condividono maggiormente il loro patrimonio genetico [13].

La caratteristica più evidente di un'impronta digitale è il pattern formato dall'alternanza tra creste e valli. La Figura 1.3 mostra che le creste seguono principalmente delle linee parallele, chiamate *ridge line*, che formano a loro volta un disegno detto *ridge pattern*.

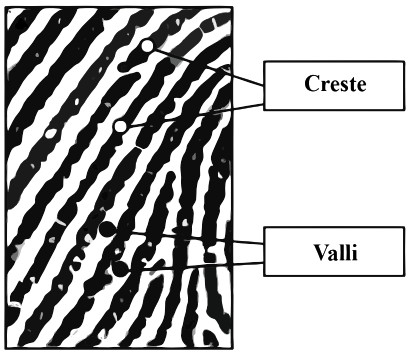

Figura 1.3: Creste e valli in un'impronta digitale

1.2.1 Persistenza e individualità

L'identificazione mediante le impronte digitali si basa su due caratteristiche fondamentali: individualità e persistenza.

L'*individualità* non è scientificamente dimostrabile, ma viene ritenuta vera sulla base di dati empirici. Questi indicano che la probabilità che le impronte di due dita distinte corrispondano è talmente ridotta da essere trascurabile [20].

Ad esempio, utilizzando il sistema di classificazione di Henry e considerando l'individuazione di 12 minuzie, la probabilità che due impronte coincidano è $P = 3,72 \times 10^{-9}$. Aumentando il numero di minuzie in esame, la probabilità decresce ulteriormente arrivando a $P = 1,45 \times 10^{-11}$ nel caso si considerino 36 minuzie.

A differenza dell'individualità, la *persistenza* è verificata scientificamente dall'anatomia. La struttura delle creste e delle valli sull'epidermide è strettamente correlata alla disposizione e allo spessore delle fibre connettive del derma, lo strato inferiore della pelle. Perciò, in caso di lesione dello strato superficiale, la pelle morta ricrescerebbe con le stesse identiche caratteristiche garantendo quindi la persistenza.

1.3 Caratteristiche

L'identificazione di una persona implica il confronto della sua impronta digitale con tutte le altre presenti all'interno di un database potenzialmente molto grande.

Nasce quindi l'esigenza di ridurre al minimo il numero di confronti con lo scopo di migliorare il tempo di risposta. La strategia consiste nell'assegnare una classe ad ogni impronta per poi confrontare i campioni sconosciuti unicamente con il sottoinsieme di quelli appartenenti alla stessa classe. Mentre la corrispondenza delle impronte digitali viene solitamente eseguita in base alle micro caratteristiche, le *minuzie*, la classificazione si basa invece su macro caratteristiche, cioè la struttura globale delle creste. Tutti gli schemi di classificazione attualmente utilizzati sono varianti del cosiddetto schema di classificazione di Henry.

All'interno dell'impronta si possono individuare due punti molto utili per la classificazione:

- *Punto di core*, cioè il punto più a nord della ridge line più interna, oppure il punto di massima curvatura della ridge line;
- *Punto di delta*, ovvero il punto in cui due creste che correvano parallele si dividono, oppure il punto in cui confluiscono creste da direzioni diverse.

Core e delta (Figura 1.4) presentano due definizioni ciascuno, poiché non sempre univocamente individuabili. Di conseguenza, qualora la prima definizione fosse difficile da applicare verrebbe utilizzata la seconda.

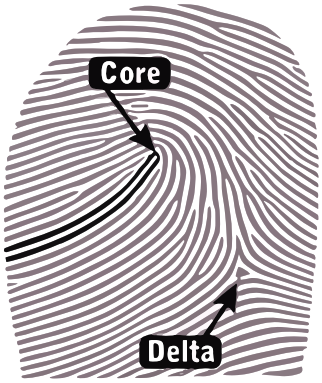

Figura 1.4: Punto di core e punto di delta in un'impronta digitale

Osservando globalmente un'impronta si può notare che, in alcune zone ad elevata curvatura, le ridge line assumono delle forme particolari, chiamate *regioni singolari*. Le regioni singolari si dividono in 3 macro categorie: loop, whorl e arch.

- *Loop*: le creste entrano da un lato, si ripiegano su loro stesse, ed escono dal medesimo lato. Si dividono in loop destri (right loop) e sinistri (left loop) in base al lato da cui entrano ed escono le creste;
- *Whorl*: le creste formano una figura chiusa circolare, ellittica oppure a spirale. Anche i whorl possono essere suddivisi in sottocategorie: plain loop, central pocket, double loop e accidental;
- *Arch*: le creste entrano da un lato, si estendono verso l'alto in prossimità del centro per poi riscendere e uscire dal lato opposto. L'ampiezza dell'angolo formato dalla piega determina la suddivisione tra plain arch, angolo meno accentuato, e tented arch, dove l'angolo è più marcato.

Ci sono svariate categorizzazioni delle regioni singolari, ma l'approccio più utilizzato è quello stabilito dal *National Institue of Standards and Technology* (NIST) [2] che considera le seguenti classi (Figura 1.5): *whorl*, *right loop*, *left loop*, *plain arch*, e *tented arch*. È importante notare che le impronte digitali non sono equamente distribuite nelle 5 classi: tented arch (2,9%), arch (3,7%), whorl (27,9%), right loop (31,7%) e left loop (33,8%).

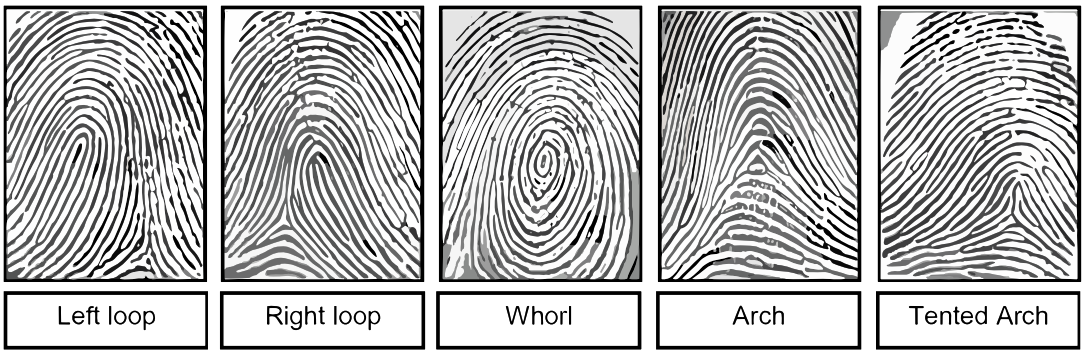

Figura 1.5: Un esempio di impronta per ognuna delle cinque classi principali

A livello locale, è possibile osservare delle frammentazioni nelle creste che prendono il nome di *minuzie*. Essendo punti di discontinuità, ovvero zone in cui è presente un comportamento anomalo delle creste, risultano estremamente importanti al fine di conferire alle impronte la loro unicità. La nomenclatura delle minuzie deriva direttamente dalla loro forma e le più comuni sono (Figura 1.6): *ridge ending*, *bifurcation*, *lake*, *independent ridge*, *point* oppure *island*, *spur* e *crossover*.

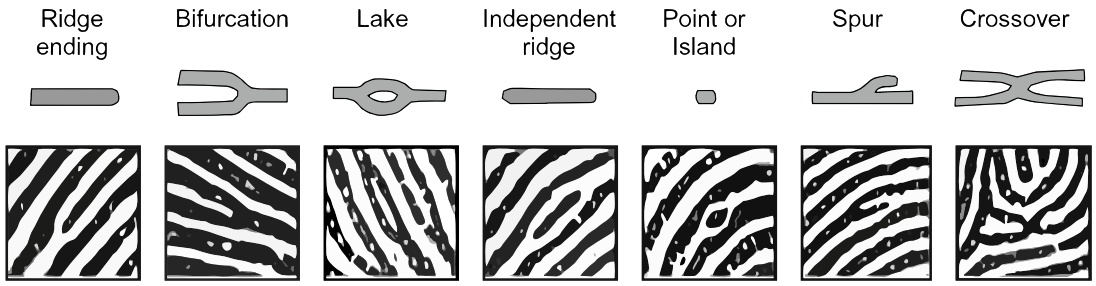

Figura 1.6: I sette tipi di minuzie più comuni

Nelle applicazioni pratiche spesso risulta estremamente difficile distinguere tutte queste tipologie, perciò vengono seguiti gli standard ISO/IEC 19794-2 [9] e ANSI/NIST-ITL 1-2011 [17] dove vengono considerate solamente le *biforcazioni* e le *terminazioni*. Poiché è stato stabilito di utilizzare unicamente due tipi

di minuzie è stato necessario definire come descriverle in modo rigoroso. Sia le biforcazioni che le terminazioni vengono descritte in termini di coordinate (x, y) e dell'angolo θ che si forma tra l'asse orizzontale e la tangente alla minuzia (Figura: 1.7).

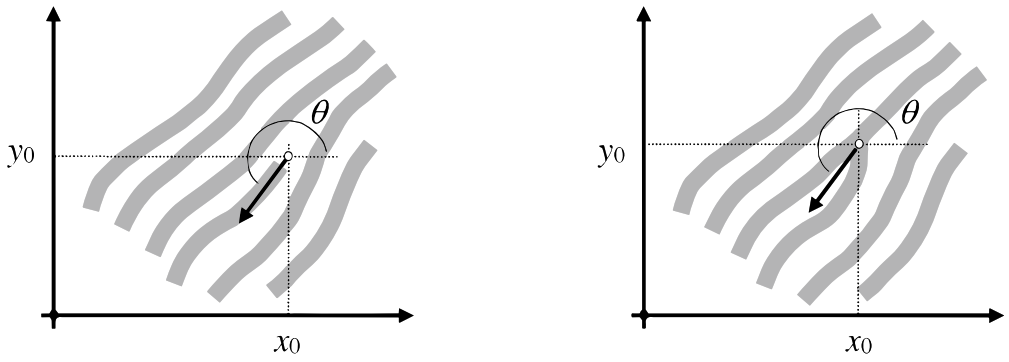

Figura 1.7: Descrizione rigorosa di una terminazione e di una biforcazione

La tipologia e il numero di minuzie sono una peculiarità altamente distintiva, motivo per il quale queste due caratteristiche vengono utilizzate per verificare la corrispondenza di due impronte.

Un'ulteriore metodologia per verificare la corrispondenza di due impronte consiste nell'individuazione del numero, della posizione e della forma dei pori della pelle all'interno delle creste. Questo metodo è uno dei più efficienti se consideriamo che per determinare univocamente l'identità di una persona è sufficiente analizzare tra i 20 e i 40 pori e un centimetro di cresta ne contiene da 9 a 18. D'altro canto è un procedimento scarsamente utilizzato perché richiede l'osservazione delle impronte ad una risoluzione estremante elevata (1000 pixel per pollice).

2

Digitalizzazione

2.1 Acquisizione delle impronte

L'acquisizione di un'impronta digitale può avvenire sia attraverso processi di acquisizione *offline* (o *inked*), che tramite processi *online* (o live-scan).

2.1.1 Inked

Nei metodi *inked* si preme il dito inchiostrato su un foglio di carta che verrà successivamente digitalizzato mediante uno scanner per documenti al fine di ottenere un'immagine digitale dell'impronta. Le immagini inked possono essere acquisite in 3 modi:

- *Rolled*: dopo aver premuto il dito inchiostrato sul foglio viene richiesto di compiere un movimento ondulatorio orizzontale per cercare di imprimere una superficie maggiore dell'impronta sul documento. Tuttavia, questo movimento può provocare deformazioni all'immagine, rendendola inutilizzabile;
- *Dab*: in questo caso, il dito non viene mosso dopo la pressione sul foglio di carta. L'immagine risultante coprirà una porzione minore dell'impronta, ma sarà meno soggetta a deformazioni;
- *Latent*: questa metodologia viene utilizzata per l'acquisizione delle impronte sulle scene del crimine. L'impronta viene estrapolata dagli oggetti toccati sfruttando i residui di grasso lasciati dalla pelle, senza contatto diretto con il dito. Anche questo caso è molto soggetto a disturbi poiché dipende strettamente dal materiale sul quale sono state impresse le impronte.

2.1.2 Live-scan

I sistemi *live-scan* permettono di catturare l'impronta attraverso la pressione diretta del dito sul sensore, il quale utilizza varie tecnologie per ottenere un'immagine. Alcuni sensori più avanzati sono in grado di ricostruire l'intera impronta combinando molteplici scansioni da diverse angolazioni.

La risoluzione delle immagini delle impronte digitali è stata standardizzata dal NIST in 500 pixel per pollice [19] e molti scanner sul mercato rispettano questa direttiva.

I diversi tipi di scanner si distinguono in base alla tecnologia utilizzata per acquisire le immagini:

- *Scanner ottici*: il sensore illumina il dito e cattura la riflessione, poiché le creste riflettono maggiormente rispetto alle valli. Tuttavia, presenza di sporco o troppa luce possono causare problemi di acquisizione;
- *Scanner capacitivi*: il sensore e il dito agiscono come due piastre di un condensatore, la capacità tra i due viene misurata e convertita in un'immagine. Sono meno sensibili allo sporco, ma possono essere influenzati da scariche elettrostatiche.;
- *Scanner termici*: il sensore rileva la differenza di temperatura tra le creste e le valli. La precisione di questi sensori è molto bassa e, poiché non soddisfano gli standard richiesti, il loro utilizzo è molto raro;
- *Sensori ad ultrasuoni*: il sensore emette degli ultrasuoni ad una specifica frequenza e ne misura l'intensità di ritorno a seguito dell'impatto con il polpastrello. La differenza di resistenza acustica della pelle e dell'aria permette di distinguere le creste (in cui si trova la pelle), dalle valli (dove è presente l'aria).

2.2 Elaborazione della scansione

Dopo aver ottenuto l'immagine dell'impronta digitale con uno dei metodi descritti nel paragrafo precedente, è necessario elaborare la scansione appena acquisita per renderla utilizzabile.

Poiché le tecniche generiche di miglioramento delle immagini non funzionano bene con le impronte digitali, sono stati sviluppati dei metodi di elaborazione specifici. In questo contesto, è importante utilizzare algoritmi che possano estrarre il maggior numero di informazioni possibili dalle immagini, al fine di rendere efficace il confronto tra le impronte.

Il processo di trattamento dell'immagine per l'estrazione delle minuzie è costituito da diverse fasi. Le principali sono illustrate nella Figura 2.1 e includono:

- *segmentazione*: distinguere l'area dell'impronta dallo sfondo;
- *stima delle orientazioni*: creare una matrice che rappresenta le orientazioni delle creste in punti specifici;
- *stima della frequenza delle creste*: individuare le porzioni dell'immagine in cui sono presenti più creste;
- *estrazione delle singolarità*: localizzare i punti di core e di delta nell'immagine;
- *ottimizzazione*: migliorare la qualità dell'immagine dell'impronta digitale;
- *binarizzazione*: convertire la scansione in un'immagine binaria, ovvero con solo due valori possibili per ogni pixel;
- *assottigliamento*: trasformare l'immagine binaria riducendo lo spessore delle creste a un solo pixel, ottenendo lo scheletro dell'impronta;
- *localizzazione*: scansionare lo scheletro per individuare i pixel corrispondenti alle minuzie.

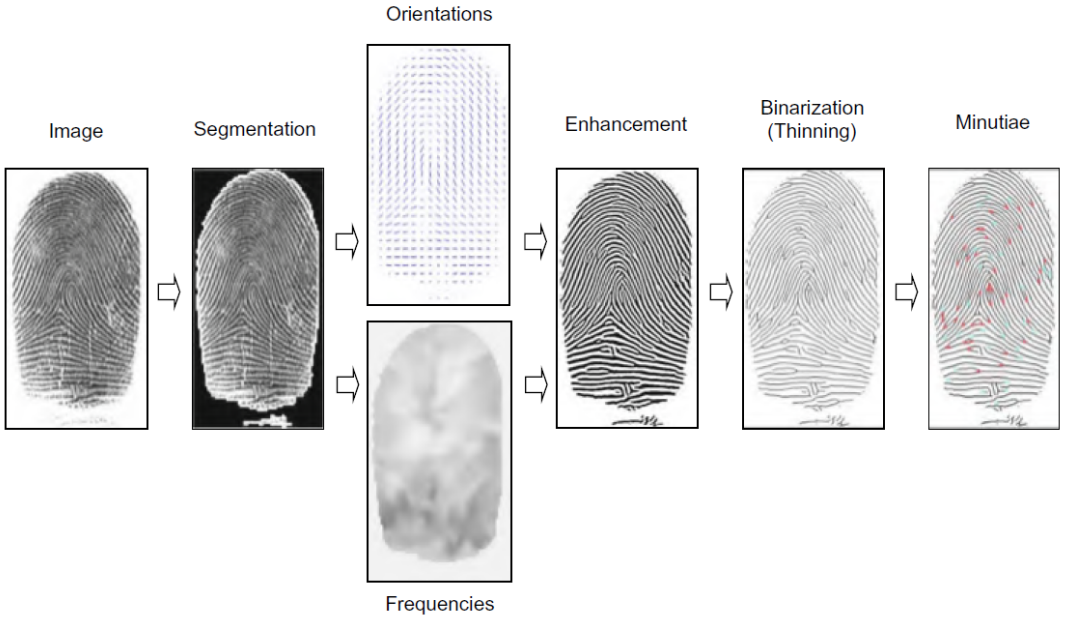

Figura 2.1: I principali passaggi dell'elaborazione di un'impronta digitale

La ricerca in questo ambito è molto attiva e perciò ogni fase menzionata può essere realizzata con molteplici tecniche, selezionate in base alla loro efficacia nel contesto specifico. Nei paragrafi successivi verranno trattati solo i passaggi utili alla classificazione, evidenziando le tecniche più comuni per ciascun passaggio del processo di elaborazione dell'immagine [16].

2.2.1 Segmentazione

Il primo passo consiste nella *segmentazione*, ovvero nell'individuare l'area dell'immagine in cui è presente l'impronta, separandola dallo sfondo circostante (Figura 2.2). Se lo sfondo dell'immagine fosse uniforme e più luminoso dell'impronta, la segmentazione sarebbe relativamente semplice. Tuttavia, spesso le immagini acquisite presentano del rumore, il che richiede tecniche più robuste. Per poter differenziare il primo piano dallo sfondo bisogna considerare le caratteristiche che li contraddistinguono: il primo piano presenta un pattern a strisce orientato, mentre lo sfondo un pattern isotropo, cioè privo di un ordinamento dominante. Sono stati sviluppati vari approcci per distinguere i due pattern, tra cui:

- La presenza di picchi negli istogrammi locali delle orientazioni delle creste;
- La varianza dei livelli di grigio in direzione ortogonale all'orientazione della cresta;
- La grandezza media del gradiente in ogni porzione dell'immagine;
- La varianza delle risposte del filtro Gabor;
- L'energia locale nello spettro di Fourier.

La maggior parte dei metodi soprastanti sono veloci e forniscono risultati soddisfacenti quando lo sfondo è uniforme e poco rumoroso, quindi sono quelli più utilizzati. Tuttavia, in situazioni più complesse, ad esempio di eccessivo rumore nell'immagine, è possibile utilizzare metodi più efficienti, ma che richiedono una maggiore potenza di calcolo.

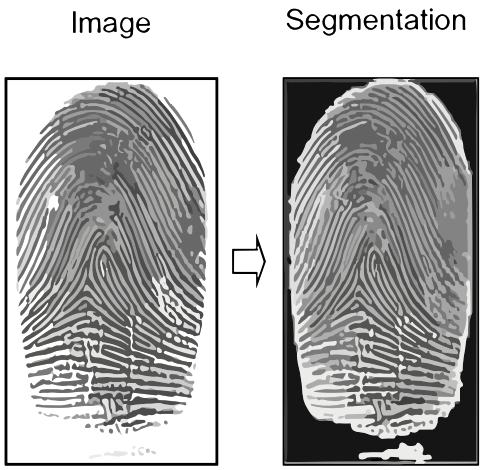

Figura 2.2: Un'impronta digitale prima e dopo la segmentazione

2.2.2 Stima delle orientazioni

Dopo la segmentazione, il passaggio successivo nell'elaborazione delle immagini delle impronte digitali è la *stima delle orientazioni locali* delle ridge line e rappresenta una delle fasi più importanti del processo.

L'orientazione locale delle creste in un dato pixel $[x, y]$ è rappresentata dall'angolo θ_{xy} che le creste formano con l'asse orizzontale quando attraversano un intorno arbitrario centrato in $[x, y]$. Poiché le creste delle impronte digitali non hanno una direzione, θ_{xy} è un angolo compreso tra [0,180°[. Calcolare l'orientazione di ogni pixel richiede molte risorse, quindi spesso viene calcolata solo in alcuni punti e i pixel mancanti vengono stimati tramite l'interpolazione.

L'*immagine direzionale* (Figura 2.3) è una matrice D i cui elementi codificano l'orientazione locale delle creste. Ogni elemento θ_{ij}, corrispondente al nodo $[i, j]$ di una griglia situata sopra il pixel $[x_i, y_j]$, indica l'orientazione media delle creste dell'impronta digitale in un intorno di $[x_i, y_j]$. Ad ogni elemento θ_{ij} viene spesso associato un valore aggiuntivo r_{ij} per indicare l'affidabilità (o la coerenza) dell'orientazione. Il valore r_{ij} è basso per le regioni rumorose e gravemente corrotte, mentre è alto per le regioni di buona qualità.

L'approccio più semplice per estrarre l'orientazione locale delle creste si basa sul calcolo del *gradiente* dell'immagine. L'angolo di fase del gradiente indica la direzione verso la quale c'è la massima variazione di intensità dei pixel. Pertanto, la direzione θ di un'ipotetica cresta che attraversa la regione centrata in $[x, y]$ è ortogonale all'angolo di fase del gradiente in $[x, y]$.

Questo metodo, sebbene semplice ed efficiente, presenta alcuni inconvenienti dovuti al fatto che la stima delle orientazioni è un'analisi di basso livello e, in quanto tale, è molto suscettibile al rumore. Di solito, per risolvere questo tipo di problema, viene effettuata una semplice media di più gradienti, ma in questo caso non è possibile a causa della natura circolare degli angoli. L'orientazione media tra 5°e 175°non è 90°, come suggerirebbe una media aritmetica, ma 0°.

Una soluzione a questo problema è stata sviluppata da Kass e Witkin [10] e consiste nel raddoppiare gli angoli in modo che ogni stima dell'orientazione sia codificata dal vettore [10]:

$$\mathbf{d} = [r \cdot \cos(2\theta), r \cdot \sin(2\theta)]$$

La media degli angoli in una finestra locale W di dimensione $n \times n$, che ci consente di ottenere una stima più robusta delle orientazioni, può essere calcolata effettuando separatamente la media per le due componenti x e y:

$$\overline{\mathbf{d}} = \left[\frac{1}{n^2} \sum_{W} r \cdot \cos(2\theta), \frac{1}{n^2} \sum_{W} r \cdot \sin(2\theta) \right]$$

Sulla base di questa intuizione, è possibile ridurre l'impatto che ha il rumore dell'immagine sull'accuratezza dell'immagine direzionale.

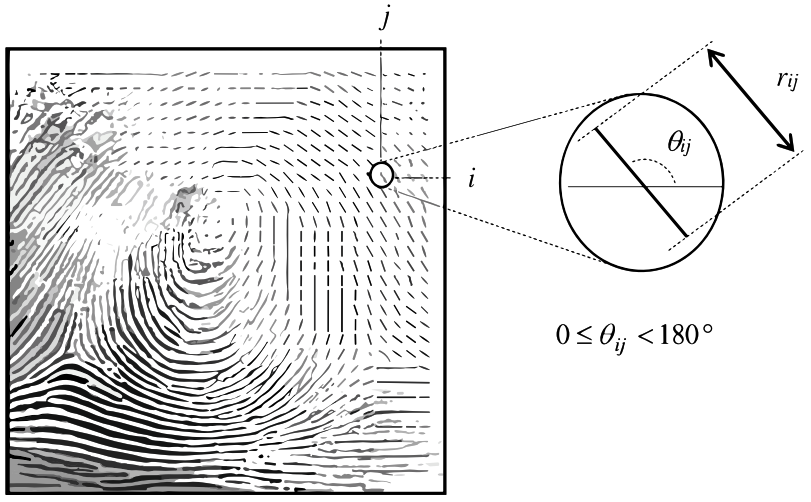

Figura 2.3: Un esempio di immagine direzionale. Ogni elemento indica l'orientazione locale della cresta in quel punto; la lunghezza del segmento indica il grado di affidabilità del valore

2.2.3 Estrazione delle singolarità

Una volta ottenuta l'immagine direzionale, l'ultimo passo per la classificazione consiste nell'*estrazione delle singolarità*. I *punti singolari*, ossia il core e il delta, sono utilizzati per classificare l'impronta digitale. Se questi punti sono posizionati in modo preciso, l'impronta può essere etichettata seguendo semplici regole che derivano dalla definizione di ciascuna classe di impronte digitali. Tuttavia, individuare questi punti singolari è un compito difficile che dipende molto dalla qualità dell'impronta e dell'immagine direzionale estratta. Come per le altre fasi, anche in questo caso esistono diverse tecniche per estrarre le singolarità, ma le più diffuse si basano sull'indice di Poincaré. Kawagoe e Tojo hanno proposto un metodo semplice ed efficace per individuare le regioni singolari nelle immagini delle impronte [11]. Sia G un campo vettoriale e sia C una curva immersa nel campo G; l'*indice di Poincaré* $P_{G,C}$ è definito come la rotazione totale dei vettori di G lungo C. Se G è il campo vettoriale discreto associato a un'immagine direzionale dell'impronta digitale D, e $[i, j]$ è la posizione dell'elemento $\theta_{i,j}$ nell'immagine direzionale. L'indice di Poincaré $P_{G,C}[i, j]$ in $[i, j]$ viene calcolato seguendo questi passi:

- La curva C è un percorso chiuso definito come una sequenza ordinata di alcuni elementi di D, tali che $[i, j]$ siano punti interni;
- $P_{G,C}[i, j]$ è calcolato sommando algebricamente le differenze di orientazione tra gli elementi adiacenti di C.
- Per poter sommare algebricamente i vettori è necessario che ad essi sia assegnata una direzione. Come abbiamo visto nel paragrafo precedente, gli elementi dell'immagine direzionale non possiedono una direzione. Una soluzione a questo problema consiste nel selezionare casualmente il senso del primo elemento e assegnare a ogni elemento successivo la direzione più vicina a quella dell'elemento precedente;
- Si dimostra che, su curve chiuse, l'indice di Poincaré assume solo uno dei valori discreti tra 0°, ±180°, ± 360°.

$$P_{G,C}(i, j) = \begin{cases} 0° & \text{se } [i, j] \text{ non appartiene a nessuna regione singolare} \\ 360° & \text{se } [i, j] \text{ appartiene ad una regione singolare di tipo whorl} \\ 180° & \text{se } [i, j] \text{ appartiene ad una regione singolare di tipo loop} \\ -180° & \text{se } [i, j] \text{ appartiene ad una regione singolare di tipo delta} \end{cases}$$

Il percorso che definisce C è la sequenza ordinata degli otto elementi d_k ($k = 0, ..., 7$) che circondano $[i, j]$. La direzione degli elementi d_k è scelta come segue: d_0 è diretto verso l'alto; d_k ($k = 1, ..., 7$) è diretto in modo che il valore assoluto dell'angolo tra d_k e d_{k-1} sia minore o uguale a 90°. L'indice di Poincaré viene quindi calcolato come segue:

$$P_{\mathbf{G},C}(i,j) = \sum_{k=0...7} \text{angle}\left(\mathbf{d}_k, \mathbf{d}_{(k+1) \bmod 8}\right)$$

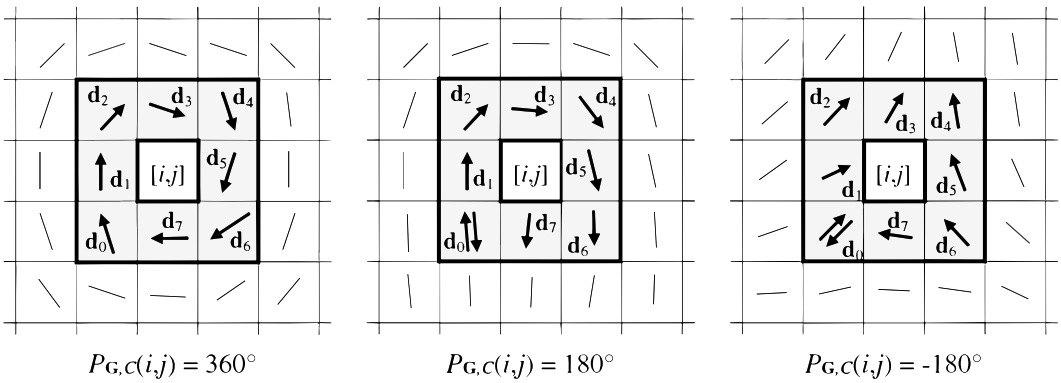

Figura 2.4: Ecco un esempio di calcolo dell'indice di Poincaré che considera un intorno di 8 punti. Le singolarità presenti sono, da sinistra verso destra, whorl, loop e delta

2.3 Tecniche di Classificazione

Nel corso del secolo scorso, il numero di impronte digitali memorizzate nei database è aumentato costantemente, rendendo necessario lo sviluppo di tecniche sempre più avanzate per classificarle. L'obiettivo di tali tecniche è quello di ridurre significativamente i tempi necessari per verificare la corrispondenza di due impronte digitali, confrontando solo quelle appartenenti alla stessa classe.

La classificazione delle impronte digitali è un problema di riconoscimento di pattern complesso, i principali fattori che determinano questa difficoltà sono:

- una ridotta variabilità interclasse e grande variabilità intra-classe dei pattern;
- la frequente presenza di rumore nelle scansioni;
- la non equa distribuzione delle impronte tra le cinque classi di classificazione.

Negli ultimi 45 anni sono stati sviluppati molti algoritmi con lo scopo di affrontare questi problemi. La maggior parte di essi utilizza l'immagine direzionale. Questa caratteristica, se calcolata con precisione, contiene tutte le informazioni necessarie per la corretta divisione in classi.

Gli approcci alla risoluzione del problema di classificazione possono essere suddivisi in diverse categorie, tra cui:

- *basati su regole*: la classe di appartenenza viene determinata seguendo alcune regole prestabilite;
- *statistici*: viene ricavato un vettore numerico da ogni impronta e successivamente si sfrutta un classificatore statistico;
- *basati su reti neurali*: si basano su percettroni multistrato che utilizzano gli elementi dell'immagine direzionale come caratteristiche di input;
- *multi-classificatori*: si combinano tra loro diversi approcci con lo scopo di ottenere un risultato più preciso possibile.

Nei prossimi paragrafi verranno trattati brevemente solo i primi due approcci, in quanto quelli di maggior interesse per quest'analisi [16].

2.3.1 Approcci basati su regole

Gli *approcci basati su regole*, anche detti *approcci basati sulle singolarità*, sono i più semplici e antichi. Un'impronta digitale può essere classificata in base al numero e alla posizione delle singolarità.

Queste tecniche sono comunemente utilizzate per la classificazione manuale e, pertanto, diversi autori hanno proposto di adottarle anche per la classificazione automatica. Infatti, hanno il vantaggio di non richiedere un set di addestramento per costruire un modello di classificazione, bensì definiscono un insieme fisso di regole per decidere la classe dell'impronta digitale.

Uno dei lavori scientifici fondamentali in quest'ambito è quello di Kawagoe e Tojo [11], che utilizzano l'indice di Poincaré per identificare il tipo e la posizione dei punti singolari e ottenere una classificazione grossolana (Tabella 2.1). Successivamente, viene tracciato il flusso della linea di cresta per ottenere una classificazione più precisa.

Classe di appartenenza	**Regione singolare**
Arch	Nessuna regione singolare
Tented arch, Loop destro e Loop sinistro	Un loop e un delta
Whorl	Due loop (o un whorl) e due delta

Tabella 2.1: Singolarità presenti nelle cinque classi più diffuse

I metodi basati sulle singolarità possono essere accattivanti per la loro semplicità, ma quando si lavora con impronte rumorose o parziali risultano inutilizzabili. Un'altra problematica è la loro efficacia limitata nell'analisi di immagini acquisite tramite scansioni live-scan, che coprono solo una porzione minore dell'impronta, rendendo difficile individuarne i punti di delta. Nelle scansioni di tipo inked, soprattutto nella versione rolled, si ricava un'immagine molto più completa.

2.3.2 Approcci statistici

Negli *approcci statistici* si crea un vettore, con dimensioni fisse, che viene popolato da caratteristiche numeriche ottenute da ciascuna impronta. Successivamente, viene impiegato un classificatore statistico per individuare la classe di appartenenza.

Tra i classificatori statistici più diffusi troviamo *la regola decisionale di Bayes*, *il k-nearest neighbor* e le *Support Vector Machines* (SVM).

Molti approcci utilizzano direttamente l'immagine direzionale come vettore di caratteristiche, semplicemente nidificandone le righe.

3

Support Vector Machine

3.1 Introduzione al machine learning

Il *machine learning* rappresenta un campo di studio all'interno dell'intelligenza artificiale che si occupa di sviluppare algoritmi in grado di apprendere dai dati e compiere predizioni o prendere decisioni. Questo processo di apprendimento si basa sulla scoperta di pattern e relazioni all'interno dei dati, senza la necessità di una programmazione esplicita [22].

Nel contesto del machine learning, esistono tre diversi tipi di apprendimento: supervisionato, non supervisionato e per rinforzo.

L'*apprendimento supervisionato* è un paradigma di apprendimento in cui un modello viene addestrato ad apprendere in autonomia dai dati forniti in input, utilizzando un insieme di dati già etichettati chiamato set di addestramento. L'obiettivo è costruire un modello che possa generalizzare e fare predizioni accurate su nuovi dati di input.

L'*apprendimento non supervisionato* si occupa di scoprire dei pattern o delle strutture nascoste all'interno dei dati di input non etichettati. In questo caso, il modello deve identificare autonomamente raggruppamenti di dati simili o strutture sottostanti in mancanza di etichette esplicite.

L'*apprendimento per rinforzo* è un paradigma in cui un agente apprende attraverso l'interazione con un ambiente. L'agente prende decisioni basate sullo stato attuale e riceve un feedback di rinforzo, positivo o negativo, in base all'azione intrapresa. L'obiettivo è quello di massimizzare una ricompensa cumulativa nel

tempo attraverso l'apprendimento di strategie di azione ottimali.

Il machine learning può essere utlizzato in diversi contesti, tre dei suoi principali campi di applicazione sono la classificazione, la regressione e il clustering.

La *classificazione* è un processo in cui il modello assegna le istanze di input a diverse classi o categorie predefinite.

La *regressione* è la previsione di un valore continuo o di una quantità numerica in base a un insieme di variabili di input.

Il *clustering* è un'attività di apprendimento non supervisionato in cui il modello raggruppa i dati in base a somiglianze intrinseche, senza etichette di classe predefinite. Questo processo aiuta a identificare pattern o raggruppamenti all'interno dei dati, consentendone una migliore comprensione.

3.2 SVM per classificare le impronte digitali

Le *Support Vector Machine* (SVM) sono modelli di apprendimento supervisionato, utilizzati per la classificazione.

Sfruttando un insieme di esempi di addestramento, ognuno dei quali è etichettato con una delle due classi possibili, SVM crea un modello che classifica i nuovi esempi scegliendo tra una delle due classi. Questo risultato è ottenuto tramite un classificatore lineare binario non probabilistico.

Un modello SVM rappresenta gli esempi come punti in uno spazio, in modo che quelli appartenenti alle due diverse categorie siano chiaramente separati da uno spazio il più ampio possibile. Successivamente, i nuovi esempi sono mappati nello stesso spazio e la previsione della loro categoria di appartenenza viene effettuata in base al lato in cui ricadono.

Le SVM si rivelano particolarmente adatte per affrontare le sfide associate alla classificazione delle impronte digitali. Una delle caratteristiche chiave delle SVM è la loro capacità di gestire efficacemente problemi di alta dimensionalità e complessità, in cui i dati possono avere un grande numero di caratteristiche o attributi.

Inoltre, le SVM sono notoriamente robuste al rumore e alle variazioni dei dati di input. Questa caratteristica è particolarmente importante nell'analisi delle impronte digitali, dove piccole variazioni, rotazioni o distorsioni possono influire sulla corretta classificazione.

3.3 Teoria delle SVM

La teoria che sta alla base del funzionamento delle SVM fu introdotta da Vapnik a partire dal 1965, nell'ambito della teoria dell'apprendimento statistico. Successivamente, Vapnik e altri studiosi ne perfezionarono ulteriormente l'approccio nel 1995 [3]. Le SVM rappresentano uno dei più diffusi strumenti per la classificazione di pattern. A differenza delle reti neurali, che stimano le densità di probabilità delle classi, Vapnik suggerì di risolvere direttamente il problema di interesse affrontando direttamente la determinazione delle superfici decisionali tra le classi.

Le SVM furono inizialmente concepite come classificatori binari, ma successivamente vennero estese per gestire la multi-classificazione. I principali tipi di SVM sono:

- *SVM lineari con pattern del set di addestramento linearmente separabili*, in cui esiste almeno un iperpiano in grado di separarli;
- *SVM lineari con pattern non linearmente separabili*. In questo caso, inevitabilmente si verificheranno degli errori di classificazione nel set di addestramento, in quanto non esiste alcun iperpiano in grado di separare completamente i pattern;
- *SVM non lineari*, in cui la superficie di separazione assume una forma complessa, senza alcuna ipotesi sulla separabilità dei pattern;
- *Estensione alla classificazione multiclasse*, in cui si combinano tra loro vari classificatori binari per ottenere un classificatore multiclasse.

Considerando due classi di pattern multidimensionali linearmente separabili, tra tutti gli iperpiani di separazione possibili, le SVM determinano quello che separa le classi con il margine massimo, chiamato iperpiano ottimo (Figura 3.1). Il margine rappresenta la distanza minima dei punti delle due classi nel set di addestramento rispetto all'iperpiano individuato.

Quando ci si trova di fronte a classi che non possono essere separate linearmente, ovvero quando non esiste un iperpiano in grado di distinguere chiaramente le classi, si ricorre all'esecuzione di un processo preliminare che prevede la mappatura dei modelli in uno spazio di dimensione superiore. Questo mapping, svolto al fine di ottenere una maggiore libertà di movimento, aumenta le probabilità di riuscire a separare le diverse classi superando le limitazioni imposte dalla linearità.

La massimizzazione del margine è strettamente correlata alla capacità di generalizzazione. Se i pattern del set di addestramento sono classificati con un ampio margine, si può ragionevolmente sperare che anche i pattern del set di test, situati nelle vicinanze del confine tra le classi, siano correttamente gestiti.

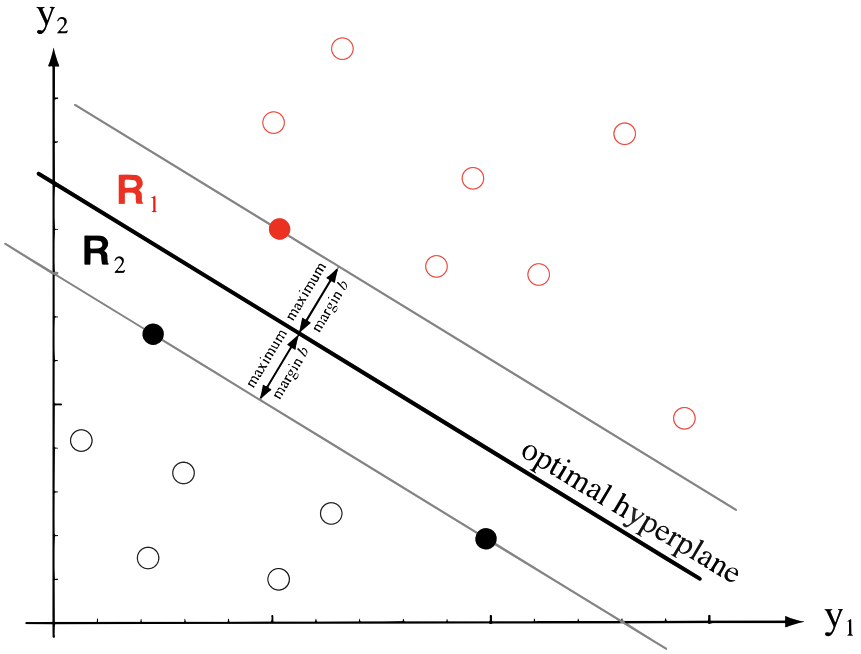

Figura 3.1: Rappresentazione grafica dell'ipepiano ottimo individuato da SVM

3.3.1 SVM LINEARI CON PATTERN SEPARABILI

Le *SVM lineari* determinano direttamente un iperpiano per separare le diverse features senza effettuare stime delle densità di probabilità dei pattern presenti nel set di addestramento [4].

Considerando due classi di pattern linearmente separabili e un set di addestramento che contiene n pattern $(x_1, y_1)...(x_n, y_n)$, dove x_i rappresenta pattern multidimensionali appartenenti a $\mathbb{R}^d$ e y_i indica le etichette delle due classi appartenenti all'insieme $\{+1, -1\}$, esistono diversi iperpiani capaci di ottenere la separazione desiderata.

Un generico iperpiano è così definito (Figura 3.2):

$$D(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$$

Dove: $\mathcal{R}_1$ e $\mathcal{R}_2$ sono le regioni delle due classi, $D(\mathbf{x}) = 0$ è il luogo dei vettori sul piano, x_p è la proiezione di x sull'iperiano, r è la distanza che separa x dall'iperpiano, w è il vettore normale all'iperpiano e $\frac{b}{||w||}$ è la distanza dall'origine, in particolare w determina l'orientazione del piano, b la sua posizione.

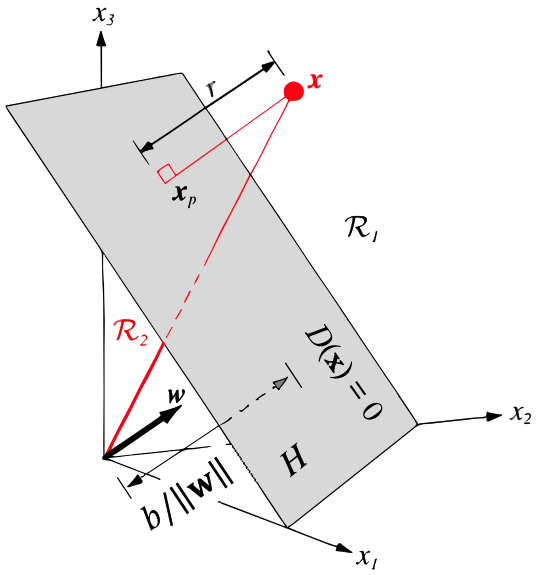

Figura 3.2: Rappresentazione grafica di un generico iperpiano tridimensionale

Quindi il vettore x può essere definito come:

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}, \quad D\left(\mathbf{x}_p\right) = \mathbf{0}$$

Si nota che $D(x)$ è legata solo alla seconda componente, e perciò:

$$D(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = r \cdot ||\mathbf{w}||$$

Di conseguenza, la distanza di x dall'iperpiano vale: $r = \frac{D(\mathbf{x})}{\|\mathbf{w}\|}$

Definiamo ora $\mathrm{D(x)} = +1$ e $\mathrm{D(x)} = -1$, due iperpiani paralleli che contengono i pattern che giacciono sul margine. L'insieme degli iperpiani $(\mathbf{w}, b)$ che separano i pattern del set di addestramento, mantenendo una distanza minima $\frac{1}{\|\mathbf{w}\|}$ su ogni lato, soddisfano, per $i = 1 \ldots n$, le seguenti equazioni:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b &\geq +1 \quad \text{se} \quad y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b &\leq -1 \quad \text{se} \quad y_i = -1 \end{aligned}$$

il vincolo di separazione diventa:

$$y_i \left[\mathbf{w} \cdot \mathbf{x}_i + b\right] \geq 1 \quad \text{per} \quad i = 1 \ldots n$$

La distanza minima tra l'iperpiano di separazione e un pattern del set di addestramento è detta margine τ.

La distanza tra i punti che giacciono sull'iperpiano $\mathrm{D(x)} = +1$ e i punti che giacciono sull'iperpiano $\mathrm{D(x)} = -1$ è il valore del margine, ovvero $\tau = \frac{2}{||w||}$.

Secondo SVM, l'iperpiano ottimo è quello che, rispettando i vincoli di separazione dei pattern, massimizza il valore di τ (oppure minimizza il suo inverso).

Abbiamo quindi un problema di ottimizzazione di questo tipo:

- Minimizza: $\frac{||w||^2}{2}$
- Vincoli: $y_i \left[\mathbf{w} \cdot \mathbf{x}_i + b\right] - 1 \geq 0 \quad \text{per} \quad i = 1 \ldots n$

I support vector, che sono i pattern del set di addestramento che si trovano sul margine (rappresentati come cerchi pieni nella figura 3.3), rappresentano i casi più difficili da classificare. Essi determinano interamente la soluzione del problema la quale può essere descritta in funzione di soli questi pattern, indipendentemente dalla dimensionalità degli spazi e dal numero di elementi nel set di addestramento.

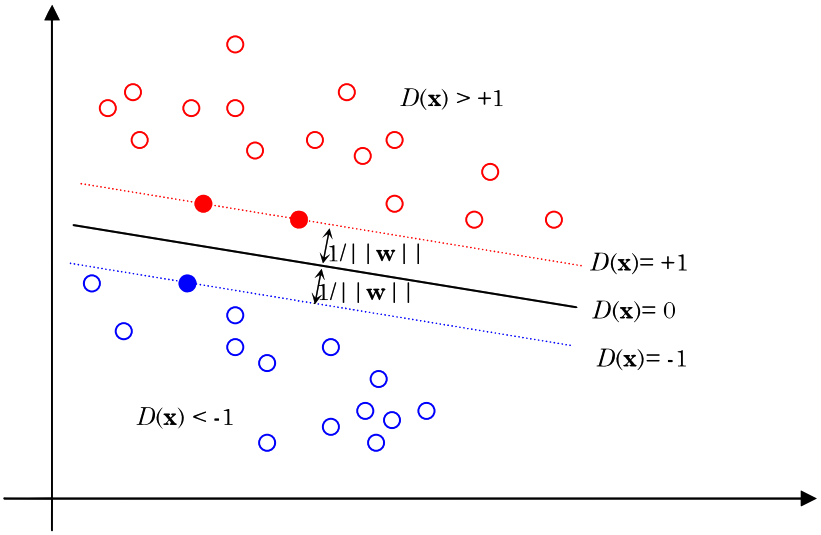

Figura 3.3: Esempio di SVM lineare con pattern separabili

Per risolvere in modo più agevole il problema di ottimizzazione precedente, si può adottare una strategia che prevede l'utilizzo della formulazione Lagrangiana seguita dalla formulazione duale [1].

La formulazione Lagrangiana consiste nell'introduzione di un moltiplicatore α_i ($\alpha_i > 0$) per ogni vincolo nella forma $equazione0$. Si procede sottraendo il vincolo moltiplicato per α_i dalla funzione obiettivo:

$$Q\left(\mathbf{w}, b, \boldsymbol{\alpha}\right) = \frac{1}{2}\mathbf{w} \cdot \mathbf{w} - \sum_{i=1}^{n} \alpha_i \cdot \{y_i \cdot [(\mathbf{w} \cdot \mathbf{x}_i) + b] - 1\}$$

da massimizzare rispetto a $\alpha_i > 0$ e minimizzare rispetto a $\mathbf{w}$ e b. Sfruttando le condizioni di Karush-Kuhn-Tucker (KKT), è possibile riformulare il problema in forma duale, esprimendo i parametri $\mathbf{w}$ e in funzione dei moltiplicatori α_i. In questo modo, la dipendenza diretta da $\mathbf{w}$ e b viene eliminata e il problema diventa la massimizzazione della nuova funzione obiettivo rispetto ai soli α_i [23]:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i \cdot \mathbf{x}_j\right)$$

Con i seguenti vincoli:

$$\sum_{i=1}^{n} y_i \alpha_i = 0 \quad \text{e} \quad \alpha_i \geq 0 \text{ per } i = 1 \ldots n$$

Grazie al passaggio in forma duale i pattern compaiono solamente come prodotti scalari. Dopo aver risolto il problema utilizzando un algoritmo di programmazione quadratica e ottenendo i valori ottimi $\alpha_1^*, \alpha_2^*, \ldots \alpha_n^*$ (le condizioni KKT affermano che $\alpha_1^* = 0$ per tutti i vettori che non sono support vector), si ottiene l'iperpiano ottimo, che è parametrizzato da:

$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i \quad \text{e} \quad b^* = y_s - \sum_{i=1}^{n} \alpha_i^* y_i \left(\mathbf{x}_i \cdot \mathbf{x}_s\right)$$

dove $(\mathbf{x}_s, y_s)$ è uno dei support vector.

La funzione distanza, che esprime la distanza dall'iperpiano è:

$$D(\mathbf{x}) = \mathbf{w}^* \cdot \mathbf{x} + b^* = \sum_{i=1}^{n} \alpha_i^* y_i \left(\mathbf{x} \cdot \mathbf{x}_i\right) + b^*$$

La funzione $D(x)$ può essere impiegata per classificare un qualsiasi pattetn $\mathbf{x}$ in base al suo segno. Per semplificare le sommatorie nella formula, è possibile considerare solamente i support vector. In un modello lineare, una volta calcolati i valori di $\mathbf{w}^*$ e b^*, non è indispensabile conservare o memorizzare tutti i support vector.

3.3.2 SVM LINEARI CON PATTERN NON SEPARABILI

In questa situazione, non tutti i modelli possono essere distinti da un iperpiano, pertanto è necessario rilassare i requisiti di separazione per consentire ad alcuni modelli, il meno possibile, di superare il confine di classe.

Per questo scopo è necessario introdurre n variabili di slack positive ξ_i, con $i = 1, \ldots n$, e adattare i vincoli di separazione:

$$y_i \left[\mathbf{w} \cdot \mathbf{x}_i + b\right] \geq 1 - \xi_i \quad \text{per} \quad i = 1 \ldots n$$

La variabile ξ_i codifica la deviazione dal margine per ogni pattern x_i del set di addestramento. Se i pattern sono separabili, le variabili di slack corrispondenti avranno valore pari a zero [8].

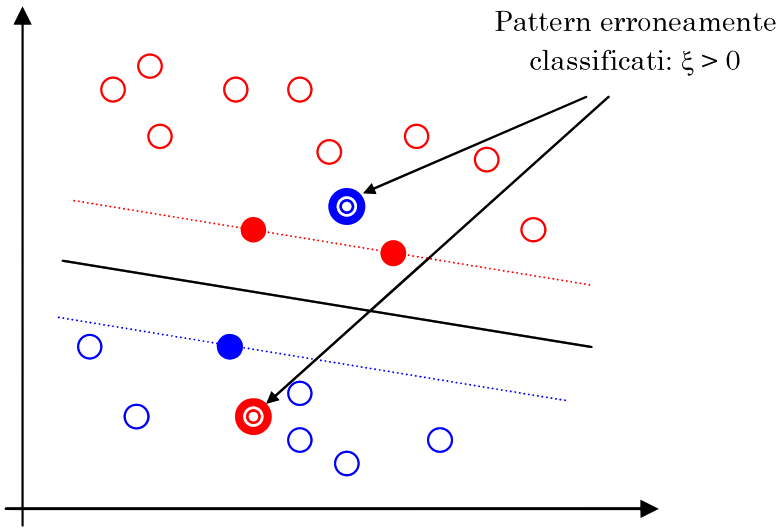

Figura 3.4: Esempio di SVM con pattern non linearmente separabili

Nel presente caso, l'iperpiano ottimale deve massimizzare il margine e contemporaneamente ridurre al minimo il numero di elementi classificati erroneamente. Di conseguenza, la funzione obiettivo e il problema di ottimizzazione subiscono le seguenti modifiche:

- Minimizza: $\frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^{n} \xi_i$
- Vincoli: $y_i \left[\mathbf{w} \cdot \mathbf{x}_i + b\right] \geq 1 - \xi_i \quad \text{per} \quad i = 1..n$

Il coefficiente C rappresenta l'importanza relativa degli errori di classificazione rispetto all'ampiezza del margine. Questo coefficiente è uno dei pochi parametri che l'utente deve scegliere per implementare le Support Vector Machines.

Nella formulazione lagrangiana ed in quella duale del problema, otteniamo un risultato quasi identico al caso in cui i dati siano linearmente separabili, con l'eccezione dell'introduzione di un limite superiore (C) per i valori dei moltiplicatori α_i:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i \cdot \mathbf{x}_j\right)$$

Con i seguenti vincoli:

$$\sum_{i=1}^{n} y_i \alpha_i = 0 \quad \text{e} \quad 0 \leq \alpha_i \leq C \quad \text{per} \quad i = 1 \ldots n$$

Il metodo di risoluzione e il modo di ottenere l'iperpiano rimangono invariati rispetto al caso di pattern linearmente separabili.

3.3.3 SVM non lineari

Le Support Vector Machine possono essere utilizzate anche per separare classi che non possono essere separate da un classificatore lineare, diversamente non sarebbe possibile applicarle a casi concreti.

La teoria inizialmente sviluppata per i casi lineari è estendibile anche ai pattern con superfici di separazioni molto complesse. Questa estensione avviene in modo semplice e intuitivo attraverso un procedimento noto come mappatura non lineare. Le coordinate degli oggetti, appartenenti all'*input space*, vengono mappate in uno spazio chiamato *feature space* utilizzando funzioni non lineari

chiamate *feature function*. Il feature space è un'area altamente multidimensionale in cui le due classi possono essere separate utilizzando un classificatore lineare. Di conseguenza, lo spazio iniziale viene trasformato nel nuovo spazio, dopodiché viene identificato il classificatore e riportato allo spazio iniziale, come mostrato nella figura 3.5.

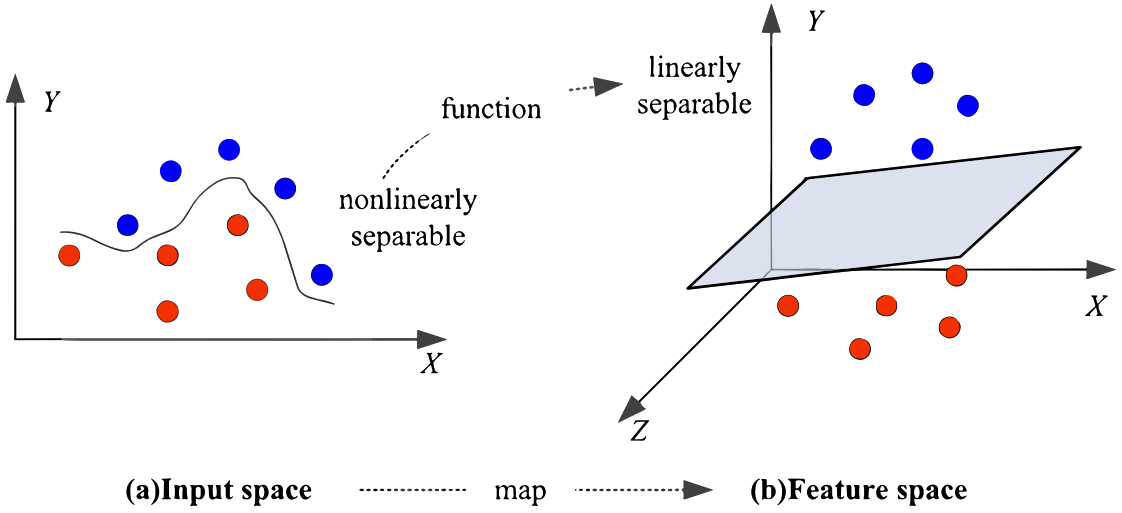

Figura 3.5: Schema della mappatura tra l'input space e il feature space

La mappatura non lineare, indicata con il simbolo Φ, trasforma i pattern dallo spazio di partenza $\mathbb{R}^d$ a uno spazio di dimensione superiore $\mathbb{R}^m$, dove $m > d$. La mappatura è formalmente definita:

$$\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^m, \quad \Phi(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots g_m(\mathbf{x})]$$

Nel contesto dello spazio $\mathbb{R}^m$, in cui si dispone di un maggior numero di gradi di libertà, i pattern $\Phi(\mathbf{x_1}), \Phi(\mathbf{x_2}) \ldots \Phi(\mathbf{x_n})$ vengono separati utilizzando le tecniche viste nei casi lineari. Ciò si traduce nell'operazione di separare i pattern $x_1, x_2 \ldots x_n$ nello spazio $\mathbb{R}^d$, utilizzando superfici di complessità arbitraria. Ad esempio consideriamo un vettore tridimensionale $X = (x_1, x_2, x_3)$, possiamo mappare questo vettore in uno spazio a 6 dimensioni utilizzando una funzione di mapping Φ:

$$\begin{array}{lll} \phi_1(X) = x_1 & \phi_2(X) = x_2 & \phi_3(X) = x_3 \\ \phi_4(X) = x_1 \cdot x_1 = (x_1)^2 & \phi_2(X) = x_1 \cdot x_2 & \phi_3(X) = x_1 \cdot x_3 \end{array}$$

La tupla nello spazio a 6 dimensioni sarà:

$$X = \left(x_1, x_2, x_3, x_1^2, x_1x_2, x_1x_3\right)$$

Esaminando la formulazione del problema duale e del lagrangiano precedentemente presentato, si può osservare che i vettori del set di addestramento compaiono solamente come prodotti scalari tra coppie di vettori. Questa caratteristica è fondamentale poiché consente di evitare la manipolazione di vettori nello spazio $\mathbb{R}^m$, che solitamente è di dimensioni molto elevate.

Infatti, mediante l'utilizzo di opportuni mapping Φ, è possibile ridurre il prodotto scalare di due modelli mappati nello spazio $\mathbb{R}^m$ a una funzione K, chiamata *funzione kernel*, dei due modelli originali nello spazio $\mathbb{R}^d$.

$$\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}') = K(\mathbf{x}, \mathbf{x}')$$

La risoluzione del problema di ottimizzazione non presenta difficoltà significative rispetto al caso lineare. Una volta che gli α_i^* sono stati individuati, la superficie di separazione può essere espressa come:

$$D(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i) + b^*$$

Per dimostrare l'importanza della scelta di un kernel appropriato prendiamo in considerazione l'esempio della tabella 3.1. Si tratta di un piccolo set di dati composto da 15 oggetti, ognuno con due parametri che appartengono a due classi, indicate come $+1$ e -1. Nella figura 3.6, la classe $+1$ sarà rappresentata da un simbolo +, mentre la classe-1 sarà rappresentata da un punto nero. L'iperpiano individuato dalle SVM sarà rappresentato da una linea continua. I vettori di supporto saranno circondati da cerchi per facilitarne l'individuazione, mentre il margine che delimitano sarà tracciato con una linea tratteggiata.

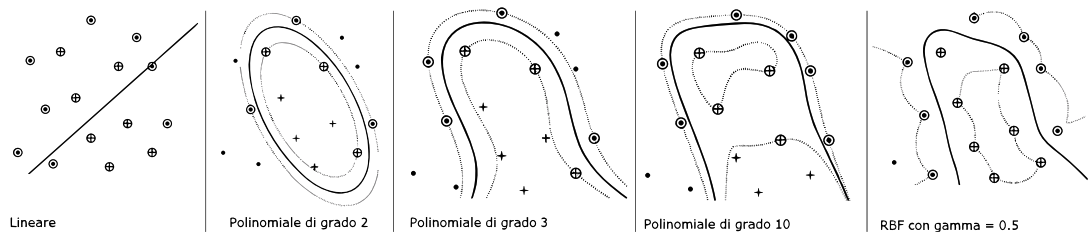

Figura 3.6: Differenza tra varie funzioni kernel applicate ai dati di esempio

Pattern	x_1	x_2	Class
1	2	4.5	1
2	2.5	2.9	1
3	3	1.5	1
4	3.6	0.5	1
5	4.2	2	1
6	3.9	4	1
7	5	1	1
8	0.6	1	-1
9	1	4.2	-1
10	1.5	2.5	-1
11	1.75	0.6	-1
12	3	5.6	-1
13	4.5	5	-1
14	5	4	-1
15	5.5	2	-1

Tabella 3.1: Dati di esempio

Osservando la figura 3.6 è evidente che, a differenza degli altri 4, il kernel lineare non risulta affatto idoneo per questo esempio. Tuttavia, si può notare che le soluzioni differiscono notevolmente tra loro. Pertanto, diventa importante avere un set di dati di prova che permetta di selezionare la configurazione migliore, al fine di evitare ciò che comunemente viene chiamato *overfitting*. Questo termine indica che l'algoritmo si adatta eccessivamente ai dati con cui è stato addestrato, ma non riesce a generalizzare il problema.

Inoltre, si può osservare che, a eccezione del kernel lineare, stiamo parlando non di funzioni semplici, ma di famiglie di funzioni che dipendono da un certo numero di parametri, comunemente noti come *iperparametri*. Questo aspetto, da un lato, ci dà maggiori speranze nel trovare la soluzione ottimale, ma, dall'altro, complica la nostra ricerca poiché dobbiamo identificare il kernel con i migliori parametri.

Di conseguenza, nel caso in cui il dataset sia estremamente limitato, è poco consigliabile impostare gli iperparametri in modo eccessivamente accurato; al contrario, è preferibile utilizzare valori predefiniti, noti per fornire una buona prestazione nel contesto specifico. Viceversa, se il dataset è di dimensioni considerevoli, è possibile apportare delle modifiche agli iperparametri con una ragionevole certezza che l'overfitting non si verifichi.

Vediamo ora alcuni esempi delle funzioni kernel più utilizzate:

- *Polinomio di grado q* (iperparametro): Le componenti $g_i(\mathbf{x}), i = 1..m$ sono ottenute come tutte possibili combinazioni di elevamento a potenze $\leq q$ delle componenti di $\mathbf{x}$.
$$K(\mathbf{x}, \mathbf{x}') = [(\mathbf{x} \cdot \mathbf{x}') + 1]^q$$
- *Radial Basis Function* (RBF) di ampiezza σ (iperparametro) :
$$K(\mathbf{x}, \mathbf{x}') = e^{\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)}$$
- *Rete neurale a due strati*:
$$K(\mathbf{x}, \mathbf{x}') = \tanh(v(\mathbf{x} \cdot \mathbf{x}') + a)$$
v ed a sono gli iperparametri, una possibile scelta è: $v = 2, a = 1$. Il numero di unità nascoste e i pesi sono determinate da SVM in autonomia.

3.3.4 SVM MULTICLASSE

Le Support Vector Machines sono comunemente utilizzate per problemi di classificazione binaria, ma spesso ci troviamo di fronte a problemi con più classi. Pertanto, è necessario adattare le SVM, che sono intrinsecamente addestrate per una sola classe, al contesto multiclasse. Esistono diverse tecniche, ma le due più famose in letteratura scientifica sono:

- *One vs One*: con questa tecnica, addestriamo una SVM per ogni possibile coppia di classi. Se abbiamo s classi nel problema, addestreremo $s \cdot \frac{(s-1)}{2}$ classificatori binari, considerando tutte le coppie possibili indipendentemente dall'ordine. Durante la fase di classificazione, il pattern $\mathbf{x}$ viene valutato da ciascun classificatore binario, che assegna un voto alla classe, tra le due, più probabile. Alla fine, il pattern $\mathbf{x}$ viene assegnato alla classe che ha ottenuto il maggior numero di voti.
- *One vs All*: con questa tecnica, addestriamo una SVM per ogni classe i del nostro problema multiclasse. Ciascuna SVM avrà due classi: w_i e $\overline{w_i}$ (non w_i). La prima conterrà gli elementi che effettivamente appartengono alla classe i, la seconda conterrà invece tutti i restanti elementi. Grazie a questa suddivisione, otteniamo così la funzione $D_i(\mathbf{x})$, che indica quanto il pattern $\mathbf{x}$ è distante dalla superficie decisionale. Maggiore è il valore della funzione, maggiore sarà la nostra sicurezza nell'appartenenza di $\mathbf{x}$ alla classe in esame. Al termine del processo di addestramento, assegniamo il campione $\mathbf{x}$ alla classe per cui la distanza dalla superficie decisionale è massima.

L'approccio One vs One è tipicamente più accurato poiché vengono addestrati un numero maggiore di classificatori; tuttavia, questo comporta un maggior costo computazionale. Per questo motivo, se il numero di classi è estremamente elevato, l'approccio One vs All diventa una scelta obbligata.

Naturalmente, esistono altri metodi più complessi per gestire la classificazione multiclasse, ma abbiamo fornito una panoramica delle due tecniche più semplici e classiche.

4 Implementazione

In questo capitolo analizzeremo un'implementazione di un classificatore di impronte digitali realizzato utilizzando SVM. Il linguaggio di programmazione scelto è *Python*, il più utilizzato nell'ambito del machine learning grazie ad una vasta gamma di librerie ben sviluppate ed estremamente popolari.

4.1 Dataset

Il dataset utilizzato è il *NIST 8-Bit Gray Scale Images of Fingerprint Image Groups (FIGS)* [24] composto da 4000 impronte digitali di dimensioni 512×512 pixel già classificate nelle 5 categorie: whorl, right loop, left loop, plain arch, e tented arch.

Per esigenze computazionali le immagini sono state ridotte ad una dimensione di 128×128 pixel [1]. Il dataset è composto dalle scansioni delle impronte in formato *png* e da file di testo in formato *txt*, i quali contengono le seguenti informazioni:

- *il genere* della persona a cui appartengono le impronte, rappresentato attraverso la lettera M (per gli uomini) e F (per le donne);
- la *classe* a cui appartengono le impronte. La struttura è la seguente: plain arch (A), left loop (L), right loop (R), tented arch (T) e whorl (W).

[1] Il dataset è accessibile al seguente URL: `https://cainvas-static.s3.amazonaws.com/media/user_data/cainvas-admin/dataset_HFu5SVU.zip`

I file di testo e le immagini presentano lo stesso nome, eccetto per l'estensione, per poterli abbinare correttamente.

4.2 Il classificatore

Il codice inizia importando le librerie necessarie per l'esecuzione del programma:

- *os* per l'accesso alle funzionalità del sistema operativo;
- *random* per la generazione di numeri casuali;
- *cv2* per l'elaborazione delle immagini;
- *numpy* per la manipolazione di array e matrici;
- *pandas* per la gestione dei dati in formato tabellare;
- *seaborn* e *matplotlib.pyplot* per la visualizzazione dei dati;
- *train_test_split* per la divisione dei dati in set di addestramento e di test;
- *SVC* per l'implementazione di un classificatore usando i support vector;
- *accuracy_score* per calcolare l'accuratezza del modello predittivo;
- *joblib* per il salvataggio e il caricamento del modello addestrato.

Successivamente, il dataset viene scaricato e decompresso. Il codice prosegue con un ciclo che analizza il contenuto della cartella su cui risiedono i dati, per poter acquisire le informazioni necessarie. Se il file ha estensione *txt*, viene aperto in modalità lettura e viene elaborato estraendo le informazioni rilevanti: il genere, il nome e il percorso del file immagine. Questi dati verranno successivamente utilizzati per creare un dataframe *pandas*, il quale ci permetterà di utilizzare agevolmente il dataset all'interno del classificatore. Infine, viene stampata l'intestazione del dataframe appena creato producendo l'output visibile nella tabella 4.1.

```
1 import os
2 import random
3 import cv2
4 import numpy as np
5 import pandas as pd
6 import seaborn as sns
7 import matplotlib.pyplot as plt
8 from sklearn.model_selection import train_test_split
```

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
import joblib

!wget -N https://cainvas-static.s3.amazonaws.com/media/user_data/
    cainvas-admin/dataset_HFu5SVU.zip
!unzip -qo dataset_HFu5SVU.zip
dir = 'dataset'

classe = []
nome_file = []
percorso_file = []
genere = []
for file in os.listdir(dir):
    if file.endswith('.txt'):
        with open(os.path.join(dir, file), 'r') as t:
            content = t.readlines()
            genere.append(content[0].rsplit(' ')[1][0])
            nome_foto = content[2].rsplit(' ')[1][:-4] + '.png'
            percorso_file.append(os.path.join(dir,nome_foto))
            nome_file.append(nome_foto)
            classe.append((content[1].rsplit(' ')[1][0]))
df = pd.DataFrame()
df['PERCORSO FILE'] = percorso_file
df['NOME FILE'] = nome_file
df['CLASSE'] = classe
df['GENERE'] = genere
df.head()
```

Codice 4.1: Importazione librerie e acquisizione dei dati

INDICE	PERCORSO FILE	NOME FILE	CLASSE	GENERE
0	dataset/s1498_03.png	s1498_03.png	T	M
1	dataset/f0976_09.png	f0976_09.png	A	M
2	dataset/f1335_08.png	f1335_08.png	T	M
3	dataset/f1664_03.png	f1664_03.png	R	M
4	dataset/f0421_01.png	f0421_01.png	W	M

Tabella 4.1: Le prime cinque righe del dataframe

Dopo aver acquisito i dati, vengono creati due grafici per poter visualizzare in modo ottimale la composizione del dataset. Il primo evidenzia il numero di campioni per ogni classe, il secondo effettua la medesima suddivisione metten-

do in luce anche il genere 4.1. A causa di un'evidente disparità tra i campioni maschili e quelli femminili è preferibile eliminare le informazioni relative al genere.

```
fig, axes = plt.subplots(1, 2, figsize=(15, 5))
sns.countplot(ax=axes[0], data = df, x = 'CLASSE')
sns.countplot(ax=axes[1], data = df, x = 'CLASSE', hue = 'GENERE')
df['CLASSE'].value_counts()
df.drop(columns='GENERE', inplace=True)
```

Codice 4.2: Creazione dei grafici ed eliminazione dei riferimenti al genere dell'individuo

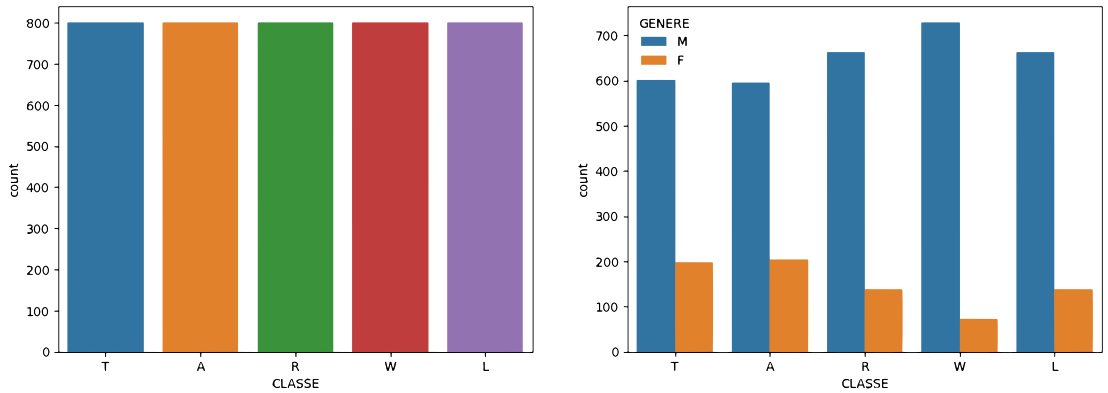

Figura 4.1: La distribuzione dei campioni nelle cinque categorie

Si procede con la mappatura delle classi verso numeri interi, operazione richiesta quando si lavora con algoritmi di machine learning o di data analysis. Infatti, per poter utilizzare le classi come input in tali algoritmi, è necessario convertire le etichette categoriche, in valori numerici. Proseguo creando una figura 4.2 che mostri l'immagine di un'impronta per ogni classe e la sua relativa etichetta.

```
classes = list(np.unique(classe))
print(classes)
map_classes = dict(zip(classes, [t for t in range(len(classes))]))
print(map_classes)
df['ETICHETTA'] = [map_classes[i] for i in df['CLASSE']]
df = df.sample(frac = 1)
df.to_csv('dataset.csv')
df.head()
dim = len(classes)
fig,axes = plt.subplots(1,dim)
```

```
fig.subplots_adjust(0,0,2,2)
for idx, i in enumerate(classes):
    dum = df[df['CLASSE'] == i]
    random_num = random.choice(dum.index)
    label = df.loc[random_num]['CLASSE']
    axes[idx].imshow(cv2.imread(df.loc[random_num]['PERCORSO FILE']))
    axes[idx].set_title("CLASSE: "+label +"\n" +  "ETICHETTA: "+str(
   map_classes[label]))
    axes[idx].axis('off')
```

Codice 4.3: Assegnazione delle etichette e creazione della figura riassuntiva

Figura 4.2: Un campione per ogni classe con la relativa etichetta

Dopo aver terminato la preparazione dei dati è possibile procedere con la creazione di un classificatore SVM utilizzando le immagini presenti nel dataframe.

Inizialmente, vengono definite tre funzioni ausiliarie che consentono di ridimensionare le immagini e di inserirle di un array numpy in modo tale da facilitare l'elaborazione.

Successivamente, vengono assegnati i dati delle immagini e delle etichette rispettivamente alle variabili *X_data* e *Y_data* , utilizzando le colonne corrispondenti del dataframe.

Viene poi eseguita la suddivisione dei dati in set di addestramento e di test utilizzando la funzione *train_test_split*. La suddivisione avviene in modo casuale, con l' 1% dei dati destinati al set di test. La distribuzione delle etichette viene mantenuta in modo uniforme tra i due set utilizzando il parametro *stratify = y_data*.

Per poter addestrare il dataset con differenti dimensioni viene definita una lista, chiamata *image_sizes*, contenente le dimensioni delle immagini. I passaggi successivi saranno eseguiti più volte, una per ogni dimensione.

Le immagini di addestramento vengono caricate e ridimensionate. Viene quindi creato un oggetto SVC (Support Vector Classifier), sfruttando il kernel

di deafult ovvero *RBF*, e viene addestrato sulle immagini di addestramento e le etichette corrispondenti utilizzando il metodo *fit*.

In seguito, le immagini di test subiscono la stessa elaborazione di quelle di addestramento. Il modello viene utilizzato per effettuare previsioni sulle immagini di test utilizzando il metodo *predict*. Viene quindi calcolata l'accuratezza delle previsioni confrontando le etichette di test originali con le previsioni effettuate sfruttando la funzione *accuracy_score*.

Infine, viene stampata l'accuratezza per la dimensione corrente dell'immagine e viene creato un grafico riassuntivo.

```
def resize_image(image_path, image_size):
    image = cv2.imread(image_path)
    resized_image = cv2.resize(image, image_size)
    return resized_image

def images_to_array(images):
    return np.array(images).reshape(len(images), -1)

def load_and_resize_images(image_paths, image_size):
    images = []
    for path in image_paths:
        resized_image = resize_image(path, image_size)
        images.append(resized_image)
    return images_to_array(images)

X_data = df['PERCORSO FILE'].values
Y_data = df['ETICHETTA'].values

X_train, X_test, Y_train, Y_test = train_test_split(X_data, Y_data,
    shuffle=True, test_size=0.01, stratify=Y_data)

image_sizes = [(8, 8),(16,16),(32,32),(64,64),(128,128)]

print(f"Numero totale di immagini: {len(X_data)}")
print(f"Numero immagini usate per l'addestramento: {len(X_train)}")
print(f"Numero immagini utilizzate per il test: {len(X_test)}")

accuracies = []

for image_size in image_sizes:
    X_train_images = load_and_resize_images(X_train, image_size)
    svm = SVC()
```

```
32      svm.fit(X_train_images, Y_train)
33      X_test_images = load_and_resize_images(X_test, image_size)
34      y_pred = svm.predict(X_test_images)
35      accuracy = accuracy_score(Y_test, y_pred)
36      accuracies.append(accuracy)
37      print(f"Accuratezza (dimensioni {image_size}): {accuracy}")
38
39  joblib.dump(svm, 'fingerprint_svm.pkl')
40
41  plt.plot(image_sizes, accuracies, 'o-')
42  plt.xlabel('Dimensioni immagine')
43  plt.ylabel('Accuratezza')
44  plt.title('Accuratezza in base alle dimensioni dell\'immagine')
45  plt.show()
```

Codice 4.4: Classificatore

4.3 Risultati

Il classificatore presentato, seppur funzionante, non ottiene risultati tali da renderlo affidabile. Ciò è dovuto ad un'estrema semplicità dell'algoritmo e ad un'elevata complessità del dataset. Per aumentare l'accuratezza si potrebbero effettuare svariate operazioni, come ad esempio:

- *pre-elaborazione delle immagini*: apportando le seguenti modifiche, quali correzione del bilanciamento del colore, normalizzazione del contrasto, eliminazione degli spazi bianchi e riduzione del rumore;
- *estrazione di feature più informative*: applicando dei filtri che consentano di ottenere dall'immagine un maggior numero di informazioni rilevanti;
- *ottimizzazione dei parametri*: esplorando diverse configurazioni di kernel (lineare, polinomiale, RBF) e valutando l'impatto di parametri come la costante di regolarizzazione C;
- *utilizzo della validazione incrociata*: invece di utilizzare una sola divisione dei dati in set di addestramento e di test, si effettuano più addestramenti avendo cura di variare sempre gli elementi appartenenti al set di test;
- *aumento del dataset*: aumentando il numero di campioni si otterrebbe una maggiore accuratezza del classificatore.

5

Conclusioni

L'analisi condotta sui sistemi di classificazione delle impronte digitali ha evidenziato che l'elaborazione delle immagini riveste un ruolo fondamentale, soprattutto quando si tratta di scansioni di impronte digitali. Ciò avviene a causa del fatto che le informazioni che è necessario acquisire da una scansione di un'impronta si discostano significativamente da quelle utilizzate per le immagini convenzionali.

Nel contesto delle impronte digitali, ci si trova di fronte a una serie di problemi che richiedono un'attenzione particolare. Uno di questi è rappresentato dalle impronte acquisite con tecniche e dispositivi differenti tra loro. A causa di ciò, le caratteristiche delle immagini catturate possono variare significativamente. Pertanto, è essenziale applicare elaborazioni specifiche per uniformare e standardizzare le immagini al fine di ottenerne una rappresentazione coerente e comparabile.

Inoltre, la scansione delle impronte è un processo molto suscettibile a rumore. Durante la loro acquisizione, infatti, possono verificarsi distorsioni, graffi o interferenze che introducono rumore nelle immagini che influenza negativamente le successive fasi di elaborazione e analisi.

Un'altra sfida significativa è costituita dall'apparente somiglianza tra le impronte appartenenti a classi diverse, nonché dalle differenze che possono esistere tra quelle della stessa classe.

È quindi fondamentale creare un classificatore che si adatti il più possibile a tutte le problematiche riscontrabili in quest'ambito. Il principale obiettivo è

fornire una classificazione accurata e affidabile, in quanto costituisce il passo preliminare per la ricerca di corrispondenza tra le impronte.

Possiamo quindi concludere che, come in molti altri ambiti dell'ingegneria, la creazione di un classificatore richiede l'individuazione di un compromesso. In questo caso, è da ricercare tra la tolleranza alle problematiche citate e l'affidabilità della classificazione.

Bibliografia

[1] Christopher J.C. Burges. «A Tutorial on Support Vector Machines for Pattern Recognition». In: *Data Mining and Knowledge Discovery* 2.2 (giu. 1998), pp. 121–167. ISSN: 1573-756X. DOI: 10.1023/A:1009715923555.

[2] Gerald Candela et al. *PCASYS- A Pattern-Level Classification Automation System for Fingerprints*. Ago. 1995. eprint: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=900754. URL: https://www.nist.gov/publications/pcasys-pattern-level-classification-automation-system-fingerprints.

[3] Corinna Cortes e Vladimir Vapnik. «Support-vector networks». In: *Machine Learning* 20.3 (set. 1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018.

[4] Richard O. Duda, Peter E. Hart e David G. Stork. *Pattern Classification (2nd Edition)*. USA: Wiley-Interscience, 2000. ISBN: 0471056693.

[5] Henry Faulds. «On the Skin-Furrows of the Hand». In: *Nature* 22.574 (ott. 1880), pp. 605–605. ISSN: 1476-4687. DOI: 10.1038/022605a0. eprint: https://www.nature.com/articles/022605a0.pdf.

[6] Francis Galton. *Finger Prints*. Macmillan, 1892. eprint: https://galton.org/books/finger-prints/galton-1892-fingerprints-1up.pdf. URL: https://galton.org/books/finger-prints/.

[7] Nehemiah Grew. «The description and use of the pores in the skin of the bands and feet, by the learned and ingenious Nehemiah Grew, M. D. Fellow of the College of physicians and of the Royal Society». In: *Philosophical Transactions of the Royal Society of London* 14.159 (1684), pp. 566–567. DOI: 10.1098/rstl.1684.0028. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rstl.1684.0028.

[8] S.R. Gunn. *Support Vector Machines for Classification and Regression*. Project Report. Address: Southampton, U.K. 1998. URL: https://eprints.soton.ac.uk/256459/.

[9] International Organization for Standardization. *Information technology — Biometric data interchange formats — Part 2: Finger minutiae data*. Standard. 2011. URL: https://www.iso.org/standard/50864.html.

[10] Michael Kass e Andrew Witkin. «Analyzing oriented patterns». In: *Computer Vision, Graphics, and Image Processing* 37.3 (1987), pp. 362–385. ISSN: 0734-189X. DOI: https://doi.org/10.1016/0734-189X(87)90043-0.

[11] Masahiro Kawagoe e Akio Tojo. «Fingerprint pattern classification». In: *Pattern Recognition* 17.3 (1984), pp. 295–303. ISSN: 0031-3203. DOI: https://doi.org/10.1016/0031-3203(84)90079-7.

[12] Arthur Kollmann. *Der Tastapparat der Hand der menschlichen Rassen und der Affen in seiner Entwickelung und Gliederung*. Voss, 1883. URL: https://books.google.it/books?id=ejwAAAAAQAAJ.

[13] Adams Wai-Kin Kong, David Zhang e Guangming Lu. «A study of identical twins' palmprints for personal verification». In: vol. 39. 11. 2006, pp. 2149–2156. DOI: https://doi.org/10.1016/j.patcog.2006.04.035.

[14] R. Leo et al. *The Fingerprint Sourcebook*. U.S. Department of Justice, Office of Justice Programs, National Institute of Justice, 2011. eprint: https://www.ncjrs.gov/pdffiles1/nij/225320.pdf. URL: https://nij.ojp.gov/library/publications/fingerprint-sourcebook.

[15] Marcello Malpighi. *De externo tactus organo anatomica obseruatio*. apud Aegidium Longum, 1665. URL: https://books.google.it/books?id=KTNoqkVZ9ukC.

[16] Davide Maltoni et al. *Handbook of Fingerprint Recognition*. 3rd. Springer, 2022. DOI: 10.1007/978-3-030-83624-5.

[17] Kevin Mangold. *Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information ANSI/NIST-ITL 1-2011 NIST Special Publication 500-290 Edition 3*. Ago. 2016. DOI: 10.6028/NIST.SP.500-290e3. URL: https://www.nist.gov/publications/data-format-interchange-fingerprint-facial-other-biometric-information-ansinist-itl-1-1.

[18] Johann Christoph Andreas Mayer, cur. *Anatomische Kupfertafeln. nebst dazu gehörigen Erklärungen*. Decker, 1788. DOI: 10.11588/diglit.7940.

[19] R. McCabe. *Information Technology: American National Standard for Information Systems: Data Format for the Interchange of Fingerprint, Facial, & Scar Mark & Tattoo (SMT) Information*. Set. 2000. DOI: 10.6028/NIST.SP.500-245. URL: https://www.nist.gov/publications/information-technology-american-national-standard-information-systems-data-format.

[20] Sharath Pankanti, Salil Prabhakar e Anil K. Jain. «On the individuality of fingerprints». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.8 (ago. 2002), pp. 1010–1025. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2002.1023799.

[21] Johann Evangelista Purkinje. *Beobachtungen und Versuche zur Psychologie der Sinne*. Reimer, 1823. URL: https://gdz.sub.uni-goettingen.de/id/PPN720941997.

[22] Stuart Russell e Peter Norvig. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson, 2020. ISBN: 9780134610993. URL: http://aima.cs.berkeley.edu/.

[23] Filip Mulier Vladimir Cherkassky. «Support Vector Machines». In: *Learning from Data*. John Wiley & Sons, Ltd, 2007. Cap. 9, pp. 404–466. ISBN: 9780470140529. DOI: https://doi.org/10.1002/9780470140529.ch9. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470140529.ch9.

[24] Craig I. Watson. «NIST Special Database 4. NIST 8-Bit Gray Scale Images of Fingerprint Image Groups». en. In: (2008-10-16 14:10:54 2008).