6. Künstliche Intelligenz

6.1. Turing-Test

6.2. Lernen

In diesem Abschnitt besprechen wir wie man an Hand von Beispielen lernt, Objekte zu erkennen und verschiedene Dinge voneinander zu unterscheiden. Diese sogenannte *Mustererkennung* hat zahllose Anwendungen. Am vertrautesten ist uns vielleicht das Filtern von unerwünschter Werbung aus unserem E-Mail Postfach. Das E-Mailprogramm lernt durch unsere Bewertungen, welche E-Mails gelöscht werden können und welche nicht. Von großer kommerzieller Bedeutung ist das Lernen von Nutzerpräferenzen. Konzerne wie Amazon lernen aus unseren Einkäufen und den Produkten, die wir uns ansehen wofür wir uns interessieren und kann so Produkte vorschlagen, die wir wahrscheinlich auch kaufen möchten.

Das Verstehen von gesprochener Sprache, das essentiell für künstlich intelligente Assistenten wie zum Beispiel Apples Siri ist, basiert auf ähnlichen Prinzipien. Man möchte gegeben eine Folge von Tönen entscheiden welchem Wort sie am ähnlichsten sind.

In diesem Kapitel werden wir sehen, dass bereits sehr einfache Methoden, die wir kaum als „Lernen“ erkennen würden, es Computern ermöglichen mit großer Genauigkeit verschiedene Aufgaben, bei denen Objekte erkannt werden müssen, zu lösen.

6.2.1. Spamerkennung

Wir werden uns zunächst auf den einfachen Fall beschränken, in dem wir Objekte einer von zwei Kategorien zuordnen möchten. Zum Beispiel möchten wir E-Mails nach Unerwünscht (*Spam*) und Erwünscht (*Ham*) unterscheiden. Dabei gibt uns der Nutzer der Mailsoftware die Beispiele aus denen wir lernen sollen, in dem er eingehende Mails als Spam oder Ham markiert.

Um Spam von Ham zu unterscheiden, müssen wir natürlich Daten sammeln. Welche Daten man genau misst kann einen großen Einfluss auf den Erfolg unserer Klassifikation haben. Man möchte Merkmale messen, die sich bei den beiden Kategorien möglichst unterscheiden. Wenn man mehrere Merkmale betrachtet sollten sie auch möglichst

unabhängig voneinander sein. Wenn man zum Beispiel Äpfel von Bananen unterscheiden möchte, könnte man das Gewicht messen. Misst man gleichzeitig auch noch das Volumen erhält man aber nur wenig zusätzliche Informationen, weil Volumen und Gewicht voneinander abhängig sind. Außerdem müssen die Merkmale natürlich auch noch einfach zu messen sein, damit man ein praktikables Verfahren bekommt.

Am einfachsten zu messen ist die Anzahl an Spam-Mails. Hat der Nutzer zum Beispiel 1000 E-Mails empfangen und davon 600 Spam bzw. 400 als Ham markiert, können wir die Wahrscheinlichkeit für Spam empirisch abschätzen. Wir sagen die *a-priori Wahrscheinlichkeit* einer E-Mail Spam zu sein ist $p_S = 0.6$ und dementsprechend $p_H = 0.4$ für Ham.

Wenn wir nur diese Information haben, welche Antwort sollten wir geben, wenn uns jemand fragt, ob es sich bei einer E-Mail um Spam oder um Ham handelt? Mit Wahrscheinlichkeit 0.6 „Spam“ zu sagen und mit Wahrscheinlichkeit 0.4 „Ham“ klingt verlockend, ist aber nicht die beste Strategie. Stattdessen müssen wir *immer* sagen es sei Spam. Weil Spam-Mails häufiger sind als Ham, machen wir den kleinsten Fehler bei der Klassifikation, wenn wir immer „Spam“ antworten.[1] Wenn wir immer Spam sagen, liegen wir in $p_S = 60\%$ der Fälle richtig. Sagen wir mit p_S Spam und mit p_H Ham, liegen wir in

$$\begin{aligned} P(\text{Richtig}) &= P(\text{Spam}) \cdot P(\text{Wir sagen Spam}) + P(\text{Ham}) \cdot P(\text{Wir sagen Ham}) \\ &= p_S^2 + p_H^2 \\ &= 52\% \end{aligned}$$

der Fälle richtig.

Die korrekte Art zu Klassifizieren ist also immer Kategorie zu wählen, die wir für am wahrscheinlichsten halten. Wenn wir keine Daten außer der Häufigkeit haben, insbesondere also unsere Informationen von der konkreten E-Mail unabhängig sind, antworten wir immer gleich. Jetzt können wir uns aber natürlich die Mail genauer ankucken und ein paar Merkmale messen. Zum Beispiel können wir überprüfen, ob sich der Absender in unserem Adressbuch befindet oder nicht.

Nehmen wir an, der Absender ist im Adressbuch. Was haben wir gelernt? Wir schauen uns wieder alle 1000 E-Mails an und zählen in wievielen der Absender im Adressbuch ist. Wir stellen fest, dass 300 Mails von bekannten Absendern kommen, davon sind 100 Spam-Mails und 200 Ham-Mails. Bevor wir den Absendern angesehen haben, glaubten wir mit $p_S = 0.6$ daran, dass die E-Mail Spam ist. Jetzt müssen wir die neuen Informationen verwenden um unseren Glauben zu aktualisieren.

[1] Das geht natürlich davon aus, dass beide Arten von Fehlern, sowohl eine Spam-Mail als Ham zu klassifizieren als auch eine Ham-Mail als Spam zu klassifizieren, gleich schlimm sind. In der Praxis ist es natürlich viel schlimmer, wenn wir fälschlicherweise eine Ham-Mail löschen und wir sollten unsere Antworten dahingehend anpassen, dass wir einen gewichteten Fehler minimieren.

Abbildung 6.1.: Thomas Bayes (1701-1761), Entdecker von Bayes Formel.

Durch Nachzählen der Mails wissen wir die Wahrscheinlichkeit, dass eine Spam-Mail einen bekannten Absender hat, wir schreiben dafür $P(\text{Bekannt}|\text{Spam}) = 1/6$. Wir möchten $P(\text{Spam}|\text{Bekannt})$ ausrechnen, also die Wahrscheinlichkeit, dass eine Mail Spam ist, wenn sie einen Bekannten Absender hat. Dazu kucken wir uns alle Mails mit bekanntem Absender an und zählen wieviele Spam sind. 100 von 300 Mails mit bekanntem Absender sind Spam, also ist $P(\text{Spam}|\text{Bekannt}) = 1/3$. Diese Art Rechnung nennt man die Formel von Bayes:

$$P(\text{Spam}|\text{Bekannt}) = \frac{P(\text{Bekannt}|\text{Spam}) \cdot P(\text{Spam})}{P(\text{Bekannt})}.$$

Wir haben also festgestellt, dass die Wahrscheinlichkeit, dass diese spezielle E-Mail Spam ist nur $1/3$ ist, weil sie einen bekannten Absender hat. Daher klassifizieren wir sie als Ham. Man könnte den Wert $1/3$ auch durch einfaches nachzählen, ohne zu Rechnen, bestimmen. Wir wollen wissen wie groß die Wahrscheinlichkeit ist, unter den Mails mit bekanntem Absender eine Spam-Mail zu finden. Es gibt 300 solche Mails und nur 100 sind Spam, also ist die Wahrscheinlichkeit $1/3$. Man möchte aber eigentlich das Nachzählen vermeiden, weil das zu lange dauert. Für jede eingehende Mail das gesamte Postfach anzusehen ist in der Praxis nicht möglich.

Wenn wir noch weitere Merkmale messen, wird die Rechnung ein bisschen komplizierter. Testen wir zum Beispiel noch, ob die Mail ein bestimmtes Wort, wie zum Beispiel „Kredit“ enthält. Wieder müssen wir in den vom Nutzer klassifizierten Mails nachzählen. 100 Mails enthalten das Wort „Kredit“, davon sind 90 Spam und 10 Ham.

	Bekannt	Kredit	Gesamt
Spam	100	90	600
Ham	200	10	400
Gesamt	300	100	1000

Wir nehmend vereinfachend an, dass *Kommt von einem bekannten Absender* und *Enthält das Wort Kredit* unabhängig voneinander sind, wir also nicht extra nachzählen müssen wieviele Mails das Wort „Kredit“ enthalten und von einem bekannten Absender kommen, sondern sofort sagen, dass ein Zehntel der Mails von bekannten Absenders das Wort „Kredit“ enthalten (weil ein Zehntel aller Mails das Wort enthalten). Wir rechnen jetzt mit Bayes Formel:

$$\begin{aligned} P(\text{Spam}|\text{Bekannt \& Kredit}) &= P(\text{Spam}) \cdot \frac{P(\text{Bekannt \& Kredit}|\text{Spam})}{P(\text{Kredit \& Bekannt})} \\ &= P(\text{Spam}) \cdot \frac{\cdot P(\text{Bekannt}|\text{Spam}) \cdot P(\text{Kredit}|\text{Spam})}{P(\text{Bekannt}) \cdot P(\text{Kredit})} \\ &= \frac{600}{1000} \cdot \frac{\frac{100}{600} \cdot \frac{90}{600}}{\frac{300}{1000} \cdot \frac{100}{1000}} \\ &= \frac{1}{2} \end{aligned}$$

Der interessiere Leser kann verifizieren, dass genau das gleiche rauskommt, wenn wir zuerst $P(\text{Spam}|\text{Bekannt})$ ausrechnen und anschließend unsern neuen Glauben an Spam oder Ham mit der Kredit-Information aktualisieren.

Tatsächliche Spamfilter bestimmen eine große Zahl von solchen Merkmalen um die Wahrscheinlichkeit auszurechnen, dass eine Mail Spam ist. Der Lernschritt ist also das Bestimmen der Wahrscheinlichkeiten $P(M|\text{Spam})$ für jedes Merkmal M an Hand der bereits klassifizierten Mails. Der Nachdenkschritt ist dann eine Anwendung von Bayes Formel.

6.2.2. Nächste Nachbarn

Bei der Spamerkennung, haben wir nur Merkmale betrachtet, die zwei mögliche Werte haben. Entweder der Absender ist im Adressbuch, oder nicht; entweder die Mail enthält Wort X, oder nicht. In vielen Bereichen, in denen wir Werte messen, die beliebige Werte annehmen können. Zum Beispiel messen wir das Gewicht von verschiedenen Fischarten um zu entscheiden ob wir sie auf das Förderband der Fischstäbchenmaschine oder in den Abfall legen möchten. Um zu verstehen, wie man aus solchen Messwerten lernen kann, werden wir zunächst das Spamproblem von einem anderen Blickwinkel aus betrachten.

Wenn wir eine E-Mail klassifizieren wollen, testen wir eine Reihe von Eigenschaften, zum Beispiel: 1) Ist der Absender im Adressbuch? 2) Enthält die E-Mail das Wort „Kredit“?.

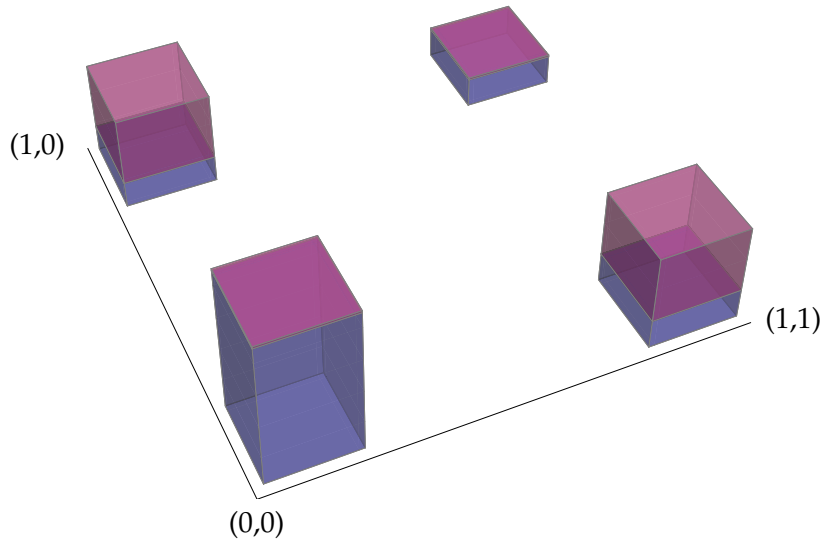


Abbildung 6.2.: Blaue Balken stehen für Spam, rote für Ham.

Weil die Antworten auf diese Fragen entweder „Ja“ oder „Nein“ sind, können wir sie mit 1 und 0 kodieren. Für eine E-Mail M können wir etwa die Ergebnisse $(0, 1)$ erhalten, d.h. der Absender ist nicht im Adressbuch und das Wort „Kredit“ kommt in M vor.

So eine Folge von Zahlen nennt man einen *Vektor*. Man kann Vektoren als Punkte im Raum verstehen. Ein Vektor mit zwei Einträgen entspricht einem Punkt in der Ebene, mit drei Einträgen entspricht er einem Punkt im dreidimensionalen Raum und mit hundert Einträgen einem Punkt im 100D-Raum.[2]

Die gleichen Tests haben wir auf allen bereits klassifizierten E-Mails auch ausgeführt. Wir haben also eine Menge von 1000 Punkten in der Ebene und für jeden Punkt wissen wir, ob es sich um eine Spam- oder eine Ham-Mail handelt. Im vorherigen Abschnitt haben wir ausgerechnet, wie wahrscheinlich es ist, dass eine Mail Spam ist, indem wir zunächst die Mittelwerte für jede Koordinate der Spam-Mails und Ham-Mail ausgerechnet und dann Bayes Formel angewendet haben. Bayes Formel funktioniert intuitiv so, dass eine Mail, die in einem Punkt liegt, in dem viele Spam-Mails liegen wahrscheinlicher Spam als Ham ist.

Im wesentlichen Rechnen wir mit Bayes Formel für einen Punkt (x, y) die Höhe der blauen und roten Balken in Abbildung 6.2 aus. Bayes Formel teilt noch durch die Höhe, weil eine mit Wahrscheinlichkeiten gerechnet wird und die Zahlen zwischen 0 und 1 sein müssen. Wenn der blaue Balken höher ist, sagen wir „Spam“, ist der rote höher, sagen wir „Ham“.

[2] Dass man sich mehr als drei Dimensionen nicht vorstellen kann, hat Mathematiker natürlich nicht davon abgehalten über die Eigenschaften von Räumen mit vielen (manchmal sogar unendlich vielen!) Dimensionen nachzudenken. Das hat sich als überaus praktisch nicht nur in der Informatik sondern auch in der Physik und den Ingenieurswissenschaften herausgestellt.

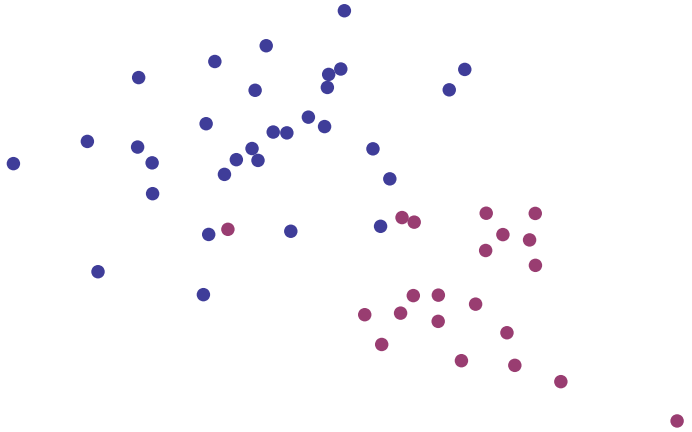


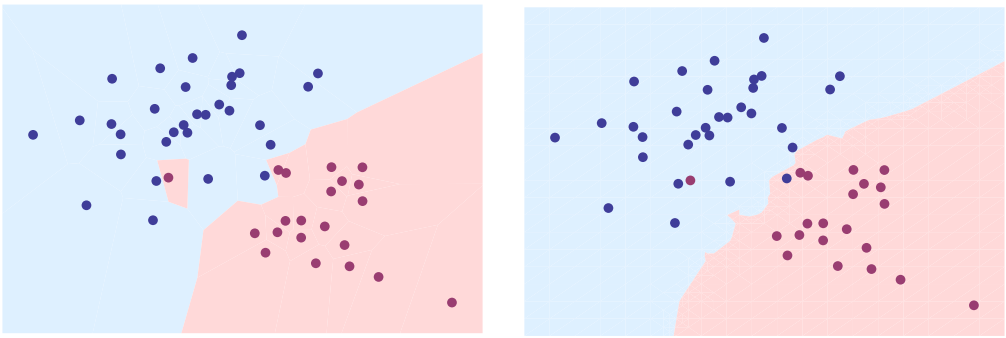
Abbildung 6.3.: Wenn Messwerte kontuinierliche Ergebnisse liefern, haben wir keine Balken wie in Abbildung 6.2, sondern Punktwolken.

Wenn wir jetzt nicht nur 0 und 1 in unseren Messwertvektor schreiben können, sondern beliebige Zahlen, ändert sich das Bild. Wir haben weiterhin für jedes klassifizierte Objekt einen Punkt, aber es ist sehr unwahrscheinlich, dass wir zwei gleiche Punkte haben. Wir erhalten eine Punktwolke wie in Abbildung 6.3.

Wenn wir jetzt eine neues Objekt bekommen und somit einen neuen Punkt in unserem Bild, wie sollten wir es am besten klassifizieren? In Abbildung 6.2 haben wir nachgezählt wieviele Spam- bzw. Ham-Emails am gleichen Punkt liegen und nach der Mehrheit klassifiziert. Es liegt nahe, dass wir uns jetzt die Nachbarschaft des neuen Punktes ansehen und an Hand seiner nächsten Nachbarn klassifizieren.

Wir rechnen also für einen neuen Punkt x die Abstände zu allen anderen Punkten aus. Jetzt können wir uns die k nächsten Nachbarn ansehen und einen einfachen Mehrheitsentscheid machen. Wieviele Nachbarn man sich am besten ansieht, also wie groß das k zu wählen ist, hängt vom genauen Problem ab. Zu klein und die Klasifikation ist nicht robust gegen Ausreißer–eine ungewöhnliche Spam-Mail sorgt dafür, dass ein paar Ham-Mails fälschlicherweise als Spam klassifiziert werden. Zu groß und die weiter entfernten Nachbarn gehören schon zur falschen Klasse. Abbildung 6.4 zeigt die Klassifikation aller Punkte der Ebene an Hand der Trainingsmenge aus Abbildung 6.3.

Dieses nächste Nachbarn Verfahren (man nennt es auch k-NN für k *nearest neighbors*), liefert recht gute Ergebnisse, hat aber eine Reihe von Nachteilen. Zunächst muss man für die Klassifikation eines neuen Punkten die Abstände zu allen anderen Punkten ausrechnen. Wenn man viele Punkte hat, und für jeden Punkt viele Messwerte, nimmt das viel Rechenzeit in Anspruch. Wichtiger ist jedoch, dass dieses Verfahren keine Abstraktion von den Trainingsdaten macht. Es lernt praktisch alle Daten auswendig und vergleicht dann nur. Wenn man etwas über die Struktur der Daten lernen möchte, zum Beispiel dass es unterschiedliche Gruppen innerhalb einer Klasse gibt, kann man dieses Verfahren dafür nicht benutzen. Im nächsten Abschnitt werden wir uns an Hand



(a) 1-NN (b) 3-NN

Abbildung 6.4.: Klassifikation der ganzen Ebene an Hand der Punkte aus Abbildung 6.3. Es wird in 6.4(a) nur der nächste Nachbarn betrachtet, in 6.4(b) werden die drei nächsten Nachbarn betrachtet.

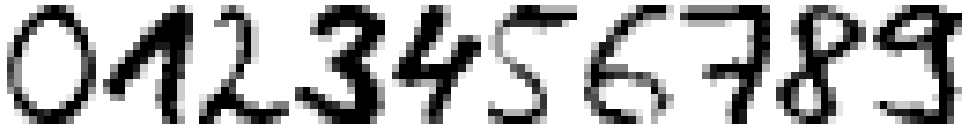


Abbildung 6.5.: Ein paar Beispiele für Ziffern aus der Datenbank handgeschriebener Ziffern.

eines Beispiels ein paar weitere einfache Verfahren ansehen, die diese Nachteile teilweise beheben.

6.3. Handgeschriebene Ziffern Erkennen

In diesem Abschnitt benutzen wir die k-NN Klassifizierung und verwandte Verfahren um handgeschriebene Ziffern zu erkennen. Wir haben eine Datenbank mit Ziffern, die auf 12×16 Pixel normiert wurden. Abbildung 6.5 zeigt ein paar der Ziffern. Die Datenbank ist in zwei Teile aufgeteilt. Eine *Trainingsmenge* von 199 Ziffern und eine *Testmenge* von 993 Ziffern. Wir trainieren unsere Erkennungsverfahren mit der Trainingsmenge und testen sie mit der Testmenge. Diese Trennung ist nötig, da sonst zum Beispiel das k-NN Verfahren perfekte Ergebnisse liefern würde–es lernt ja die gesamte Trainingsmenge auswendig.

Um die Ziffern in Punkte im Raum umzuwandeln, interpretieren wir jedes der $12 \cdot 16 = 192$ Pixel als eine Zahl zwischen 0 und 1, 0 bedeutet Weiß, 1 Schwarz, und Grauwerte werden durch Zahlen dazwischen ausgedrückt. Jede Ziffer entspricht also einem Punkt im 192-dimensionalen Raum. Wenn man nicht nur die statischen Bilder der Ziffern zur Verfügung hat, sondern auch noch Daten darüber hat, wie sich der Stift beim Zeichnen

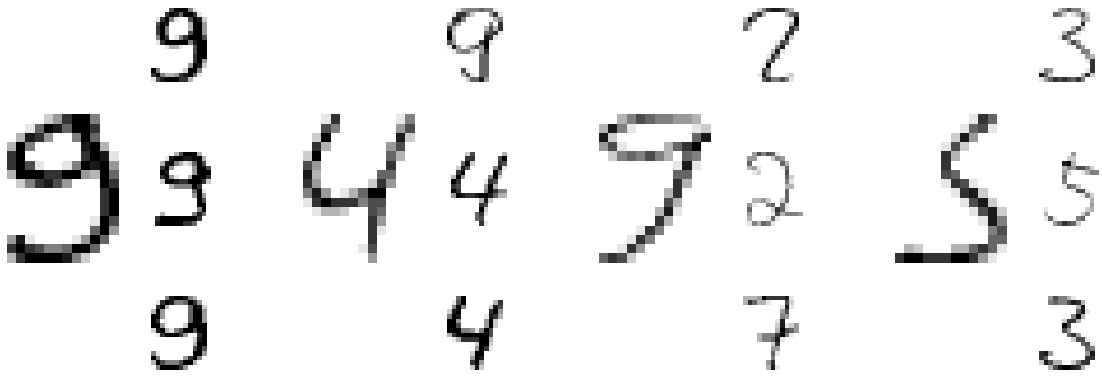


Abbildung 6.6.: Ziffern und ihre drei nächsten Nachbarn. Der nächste Nachbar ist oben. Manche der Ziffern sind auch für mich schwer zu erkennen. Die am weitesten rechts stehende Ziffer ist laut Datenbank eine 5. Meiner Meinung nach ist die 3-NN Klassifikation als 3 aber durchaus gerechtfertig.

bewegt hat, kann man sich natürlich noch weitere Merkmale überlegen, die bei der Klassifizierung hilfreich sein könnten. Zum Beispiel kann man die Winkelsumme der Stiftbewegungen bestimmen, oder die Geschwindigkeit. Die korrekte Auswahl der Merkmale, die man zum Klassifizieren verwenden möchte hat einen großen Einfluss auf den Erfolg und erfordert eine Menge Erfahrung.

Auch wie man die Distanz im Raum definiert spielt eine Rolle. Am einfachsten ist es, die euklidische Distanz zwischen den Punkten zu nehmen, also die Länge der Verbindungslinie. Eine Alternative ist die Winkeldistanz. Statt die Länge der Verbindungslinie zu messen, messen wir den Winkel zwischen den beiden Geraden, die die Punkte mit dem Ursprung verbinden. Wenn zwei Ziffern an den gleichen Pixeln dunkel sind, können sie eine große euklidische Distanz voneinander haben, wenn eine dunkler ist als die andere, der Schreiber also fester aufgedrückt hat. Die Winkeldistanz ist dann aber weiterhin gering. Für unsere Zwecke reicht es allerdings aus nur die einfache euklidische Distanz zu berechnen.

Für diese Klassifikationsaufgabe funktioniert bereits das einfache nächste Nachbarn Verfahren extrem gut. Wenn man nur den nächsten Nachbarn betrachtet, erreicht man schon mehr als 93% korrekte Ergebnisse. Betrachtet man die drei nächsten Nachbarn verbessert sich das auf fast 95%. Abbildung 6.6 zeigt ein paar Ziffern und ihre nächsten Nachbarn.

Wie aber schon im letzten Abschnitt gesagt, braucht die Klassifizierung mit diesen Verfahren recht viel Rechenzeit. Was kann man machen um das zu umgehen?

Eine einfache Idee ist es, die Anzahl der Punkte zu denen man Entfernungen ausrechnen muss zu verringern, indem man sich repräsentative Beispiele aussucht. Wenn man wie wir eine von Menschen mit den richtigen Klassen annotierte Trainingsmenge hat, kann man zum Beispiel für jede Zifferngruppe den Mittelpunkt ausrechnen. Dann

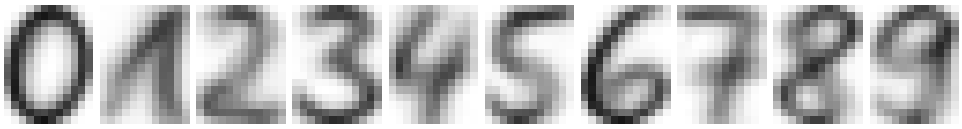


Abbildung 6.7.: Die Zentren der verschiedenen Ziffernklassen.

benutzt man diese zehn „Durchschnittsziffern“ zur 1-NN Klassifikation. Statt wie vorher die Abstände zu fast zweihundert Trainingsziffern ausrechnen zu müssen, werden dann nur noch die Abstände zu zehn Ziffern ausgerechnet. Abbildung 6.7 zeigt die Durchschnittsziffern.

Weil man eine ganze Menge Informationen verliert, ist natürlich auch die Erkennungsrate viel geringer. Lediglich 85% der Testmenge werden an Hand der Zentren korrekt klassifiziert. Besonders schlecht funktioniert diese Methode, wenn die Punkte einer Klasse im Raum nicht eine ungefähr kugelförmige Wolke bilden. Wenn sie stattdessen lang gezogen sind, oder gar aus mehreren voneinander recht unterschiedlichen Untergruppen bestehen, ist das Zentrum wenig repräsentativ. Außerdem erfordert diese Methode eine bereits von Hand klassifizierte Trainingsmenge. Eine solche zu erzeugen ist natürlich recht aufwändig.

Ein weiteres Verfahren, das im Prinzip auf der gleichen Idee basiert, kommt ohne eine vorklassifizierte Trainingsmenge aus. Bei diesem k-*Means* genannten Verfahren sucht ein Algorithmus selbstständig Repräsentanten für k Gruppen. Es funktioniert sehr einfach: Wir beginnen damit uns k zufällige Repräsentanten im Raum zu wählen. Das können entweder Punkte der Trainingsmenge sein, oder komplett ausgewürfelte. Jetzt klassifizieren wird die gesamte Trainingsmenge an Hand dieser Repräsenten. Dadurch bekommt jeder Repräsentant eine Menge von Punkten, die zum ihm näher sind als zu den anderen Repräsentanten. Wir verschieben jeden Repräsentant in die Mitte der ihm zugewiesenen Gruppe. Wir wiederholen diesen Gruppieren-Verschieben Schritt solange bis sich die Repräsentanten nicht mehr bewegen.

Diese Repräsentanten sind natürlich noch nicht klassifiziert, weil die Trainingsdaten nicht vorklassifiziert sind. Aber weil es sich nur um k Stück handelt, können wir einen Menschen bitten ihnen Klassen zuzuweisen. Anschließend können wir Testdaten an Hand der so ausgewählten Repräsentanten klassifizieren.

Im Idealfall ist jeder Gruppe in den Trainingsdaten ein Repräsentant zugeordnet. Natürlich kann es aber passieren, dass zwei oder mehr Repräsentanten in die selbe Gruppe fallen und dafür manche Gruppen gar keinen Repräsentanten abbekommen. Daher ist es in der Regel eine gute Idee k größer zu wählen als die Zahl der Gruppen, die man in seinen Daten erwartet. Abbildung 6.8 zeigt die Repräsentanten die für 17-Means gefunden werden (weil der Algorithmus randomisiert ist, kommen natürlich immer andere Repräsentanten raus). Die Erkennungsrate liegt zwischen siebzig und achtzig Prozent, je nachdem wie viel Glück man bei der Berechnung der Repräsentanten hat.

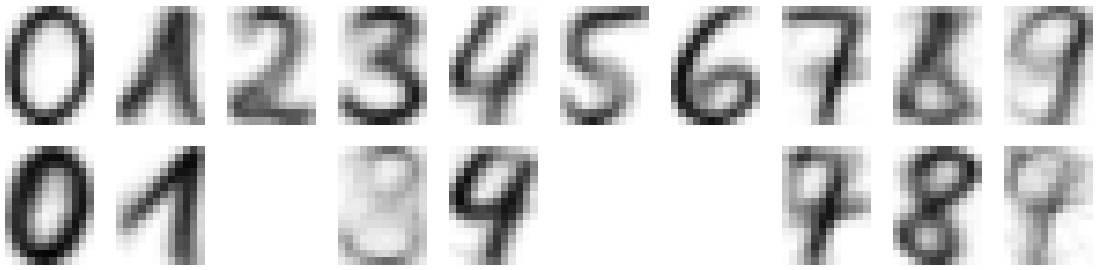


Abbildung 6.8.: Ein Beispiel für Zentren, die mit k-means automatisch gefunden werden. Man bemerke, dass es bei der 9 zwei Untergruppen gibt–die mit einem geraden Strich und die mit einem Hakenstrich. k-Means findet diese beiden Untergruppen. Ähnliches bei der 1.

Neben der Anwendung zur Klassifizierung, kann man den k-Means Algorithmus dazu benutzen um versteckte Strukturen in seinen Daten zu finden. Zum Beispiel kann man sehen, dass es verschiedene Arten gibt, die 1 zu schreiben, man gerader, mal dachförmiger. Ähnliches tritt bei der 9 auf. Manche Menschen machen einen geraden Strich, andere einen einen Haken.

Spannender ist das natürlich bei anderen Datensätzen als handgeschriebenen Ziffern. Man könnte zum Beispiel verschiedene chemische Marker an den Zellmembranen von Krebszellen messen und dann lernen, dass es mehrere Untergruppen dieser Krebsart gibt, die verschieden gut auf unterschiedliche Medikamente ansprechen.

6.4. Spiele