Riconoscimento di movimenti e posture del corpo
Report per l'Esame di Fondamenti di Machine Learning

Francesco Baraldi
132097
Corso di Laurea in Ingegneria Informatica
256745@studenti.unimore.it

Abstract

Questo documento presenta un'analisi di diversi modelli di machine learning per il riconoscimento delle attività umane (Human Activity Recognition, HAR). Lo scopo è quello di riconoscere correttamente la posizione di una persona tra 5 possibili classi, per farlo si è utilizzato un dataset che contiene i dati che quattro accelerometri, posti su quattro diverse posizioni del corpo, hanno raccolto in un esperimento di 8 ore su quattro soggetti diversi. Inoltre vegnono utilizzati dati generici sui soggetti, come il nome, peso, l'altezza, l'indice di massa corporea, l'età e il sesso. Gli accelerometri sono posti sulla vita, sulla coscia sinistra, sul braccio destro e sulla caviglia destra. Il dataset contiene più di 165 mila samples.

1 Introduzione

La ricerca nel campo dello Human activity recognition (HAR) ha diverse applicazioni, soprattutto in ambito medico, infatti può permettere alle persone più anziane, ma anche a persone più deboli o malate, di essere monitorate e in caso di caduta o incidente la tecnologia può intervenire tempestivamente, riconoscendo un'anomalia nella posizione del soggetto, consentendo quindi maggiore sicurezza.

I possibili approcci allo HAR sono due, l'**image processing** consiste nell'analizzare immagini e video per monitorare lo stato di un soggetto, questo implica per che il controllo può essere fatto solamente in determinati luoghi dove sono presenti per esempio delle telecamere adibite allo scopo specifico. Un altro approccio è basato su **sensori indossabili** come degli accelerometri, in questo caso il soggetto dovrà indossare questi accessori per poter effettuare il controllo ma non ci saranno vincoli di spazio. Con il veloce sviluppo dei dispositivi IoT (Internet of Things) questo approccio potrà essere sempre più facile da implementare, per esempio integrando dei dispositivi per il monitoraggio direttamente nei vestiti. Un possibile problema potrebbe essere la miniutirizzazione dei dispositivi hardware, e la relativa gestione energetica, infatti questi dispositivi avranno bisogno di energia per funzionare.

In questo report però ci si limita ad analizzare un problema concreto, in particolare i dati di 4 accelerometri posti sulla vita, sulla coscia sinistra, sul braccio destro e sulla caviglia destra di 4 soggetti in un esperimento durato 8 ore. Questi dati sono raccolti, con altre caratteristiche dei soggetti che hanno partecipato all'esperimento (nome, sesso, età, altezza, peso, indice di massa corporea), in un dataset di oltre 165 mila samples in totale. Lo scopo è quello di riconoscere correttamente la posizione di un soggetto con le features disponibili, quindi è un problema di classificazione e le classi sono cinque:

- **sitting-down**
- **standing-up**
- **standing**
- **walking**

- **sitting**

Il documento prosegue con la sezione 2, nella quale viene fatta un'analisi dei dati utilizzati, in particolare verranno mostrati i risultati dell'exploratory data analysis (EDA) del dataset. Nella sezione 3 vengono discussi i possibili modelli di machine learning che possono essere applicati a questo problema e descritti quelli che sono stati scelti. Inoltre per ogni modello scelto verranno presentate tre versioni: una con i dati in input grezzi, non processati, una seconda versione con in input i dati pre-processati, e infine la terza versione sarà la migliore versione tra la 1 e la 2 in ensemble.Nella sezione 4 viene descritto il processo di implementazione dei vari modelli scelti, come è stato fatto il tuning degli iperparametri, il pre-processing e il metodo di valutazione. Infine nella sezione 5 si traggono le conclusioni con un confronto tra i vari modelli.

2 Analisi dei dati

Il dataset [**?**] contiene 165633 sample e 19 features, dopo una veloce analisi dei dati è stata fatta una conversione di tipo da stringa a tipo numerico, in particolare la feature *body_mass_index* e la feature *how_tall_in_meters* sono state convertite da stringhe a tipo numerico. Inoltre la feature *z4*, che rappresenta la coordinata z del quarto accelerometro, avendo un sample con un valore non numerico è stata convertita in modo forzato in tipo numerico, trasformando il valore errato in NaN, dopodichè è stato scelto di eliminare il sample con questo valore dal dataset. Il resto del dataset non contiene valori nulli o anomali. Il dataset risulta sbilanciato per quanto riguarda la feature target, chiamata *class*, come si vede dalla figura 1 infatti sono presenti più samples per le classi *sitting*, *standing* e *walking*. Quindi sarà necessario stratificare durante lo splitting del dataset e utilizzare delle metriche opportune, come l'accuracy bilanciata invece dell'accuracy semplice.

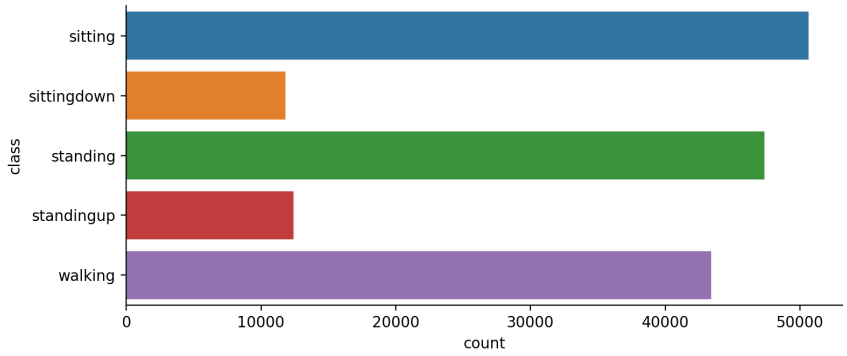


Figure 1: Bilanciamento della variabile target.

Come si vede invece dalla figura 2 i sample raccolti per ogni soggetto dell'esperimento sono bilanciati per tre soggetti mentre sono in misura minore per il soggetto *josecarlos*. Dalla figura 3 si vede invece per ogni soggetto la percentuale di sample per ogni classe, e questo rispecchia il bilanciamento generale delle classi mostrato in figura 1, infatti le percentuali sono simili, in particolare le classi *sittingdown* e *standingup* sono presenti in misura minore.

In figura 4 sono rappresentate le distribuzioni di probabilità dei dati raccolti dall'accelerometro 2, cioè quello posto sulla coscia sinistra, classificati in base alla posizione dei soggetti. Si può notare come nel caso della classe *sitting* e *standing* tutte e tre le coordinate sono più concentrate nell'intorno di un'unico valore, infatti la cosiddetta *campana* è più stretta e alta. Questo può significare che se il valore letto è lontano dal valore di riferimento difficilmente la posizione sarà una delle due tra *sitting* e *standing*.

Nella figura 5 si vede la relazione tra le coordinate dell'accelerometro 2 e le coordinate di tutti gli accelerometri divisi per classe. Le informazioni utili che si ricavano da questo grafico sono che le coordinate dell'accelerometro 3, quello sulla caviglia destra, differiscono da quelle dell'accelerometro 2 nel caso di posizione *standingup* e *walking*, infatti la zona rossa,

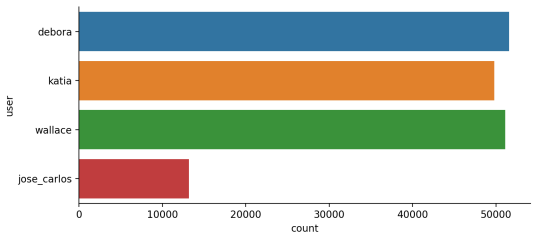


Figure 2: Bilanciamento della variabile target.

Figure 3: Bilanciamento della variabile target.

che rappresenta la classe la posizione *standingup*, assume valori minori in termini di coordinate dell'accelerometro 3. Il resto delle classi non sono distinguibili e le coordinate assumono un range di valori simile tra loro.

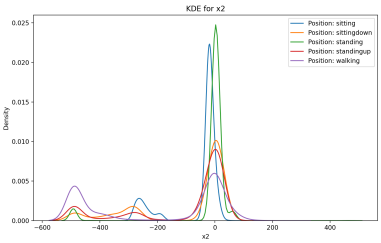
L'accelerometro 1, quello posto sulla vita, legge delle coordinate a seconda della posizione che sono rappresentate on figura 6. Come si vede dai tre grafici rappresentanti le tre coordinate, i valori non variano molto al variare della posizione, eccezion fatta per la coordinata z che diminuisce il suo valore in posizione *sitting down*. Questa informazione è ragionevole infatti la vita è la parte del corpo che cambia meno la sua posizione nello spazio al variare della postura. Quando ci siede invece l'altezza della vita (rappresentata dalla coordinata z) si abbassa e quindi diminuisce la coordinata relativa. Inoltre dai grafici 6a, 6b, 6c è possibile notare degli high leverage point presenti nei dati, cioè quei sample che differiscono dagli altri in termini di valore di una feature, in questo caso le feature *x1* *y1* e *z1*.

Nella fase di pre-processing delle variabili è stato fatto un encoding delle features categoriche, in particolare la feature *gender* è stata trasformata con un oggetto di tipo `LabelEncoder()`, che ha mappato i due possibili valori, *man* e *woman*, in numeri *0* e *1* rispettivamente. Inoltre è stato fatto l'encoding con lo stesso metodo anche della variabile target, *class*, mappando le cinque classi con numeri da *0* a *4*. Dopodiché è stato effettuato il min-max scaling alle features, che porta il valore di ogni feature tra 0 e 1, per velocizzare il training dei modelli. Inoltre per alcuni modelli è stato deciso di fare una riduzione delle features tramite la PCA (Principal Component Analysis), che sarà però approfondita nella sezione 4. Infine il criterio di splitting scelto per il dataset è stato il seguente:

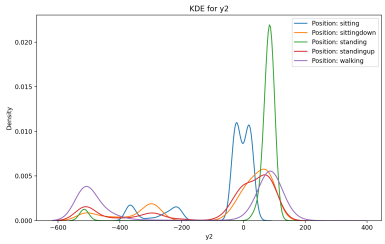
- **training set**: 80% del dataset originale
- **test set**: 20% del dataset originale

Lo splitting è stato fatto in modo stratificato in modo da effettuare la divisione mantenendo bilanciate le classi della variabile target.

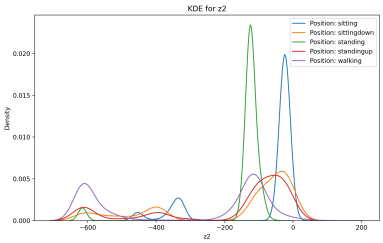
In fase di cross validazione sarà poi effettuato un ulteriore splitting del training set, in validation set e development set per fare la model selection per la scelta opportuna dei parametri



(a) Distribuzione della coordinata x.



(b) Distribuzione della coordinata y.



(c) Distribuzione della coordinata z.

Figure 4: Densità delle coordinate dell'accelerometro 2.

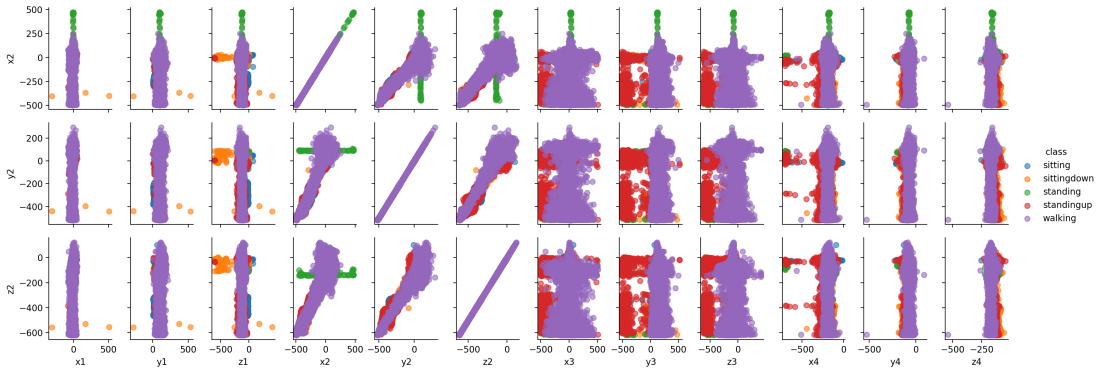


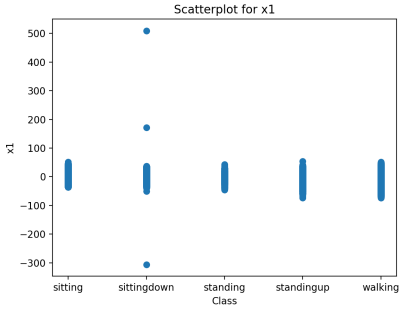
Figure 5: Pairplot delle coordinate dell'accelerometro 2.

dei vari modelli, e la model assessment per stimare le prestazioni dei modelli.

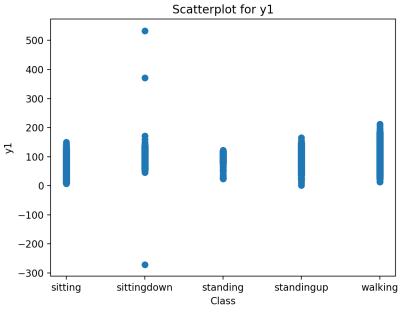
3 Discussione dei modelli

Il problema in esame è un problema di classificazione perché è necessario appunto "classificare" la postura di un soggetto tra cinque possibili posizioni. Gli algoritmi di machine learning adatti a questo tipo di problema sono diversi, i principali sono la *softmax regression*, è come la logisitc regression ma per il caso multiclasse e utilizza la funzione softmax invece della funzione logisitica; il *classificatore naive Bayes*, basato principalmente sul teorema della probabilità di Bayes; il *K-Nearest neighbors*, che assume che sample simili siano vicini nello spazio delle features, quindi calcola la distanza relativa tra le features e per ogni sample prende i K samples più vicini e ne restituisce la moda delle label. Il parametro K è quindi un iperparametro da scegliere opportunamente. Il **decision tree**, o **alber decisionale** in italiano, è un metodo di apprendimento nel quale i dati sono suddivisi in base a un certo parametro in sottoinsiemi sempre più piccoli, fino ad arrivare ad avere insiemi che non sono più divisibili, questi saranno le foglie dell'albero e definiranno la label. In particolare gli alberi decisionali sono composti da:

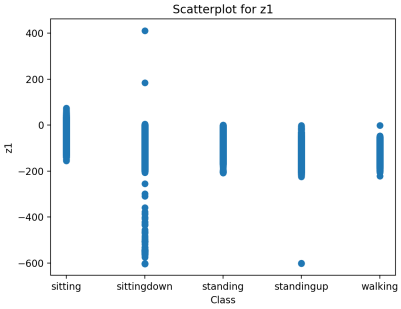
- **nodi**: controllano il valore di un determinato parametro sui dati,



(a) Distribuzione della coordinata x.



(b) Distribuzione della coordinata y.



(c) Distribuzione della coordinata z.

Figure 6: Densità delle coordinate dell'accelerometro 2.

- **rami**: collegano più nodi e rappresentano l'esito di un test fatto dal nodo padre,
- **foglie**: sono i nodi terminali e determinano il risultato finale.

Per selezionare le features su cui effettuare il test e il valore *cutpoint*, cioè il valore limite da testare, si utilizza il **recursive binary splitting** che ad ogni passo seleziona le features e il cutpoint che riducono maggiormente l'impurità dei nodi, un nodo è puro quando tutti i samples appartenenti alla medesima suddivisione hanno la stessa label. L'indice per misurare l'impurità è un iperparametro e può essere scelto, una possibile soluzione è usare l'indice di Gini, cioè la probabilità che un una variabile sia classificata in modo errato. Uno dei vantaggi degli alberi decisionali è che sono semplici da costruire e veloci, inoltre escludono in modo automatico le features non importanti, ha però lo svantaggio di tendere facilmente all'overfitting.

Un altro modello adatto a questo problema è il **support vector machine** (SVM), infatti il SVM ha lo scopo di dividere i sample in classi con un iperpiano di uno spazio n-dimensionale, dove n è il numero delle features. L'iperpiano da scegliere è quello con il massimo margine, cioè la massima distanza tra sample di classi diverse. Un aspetto fondamentale del SVM è che se i sample non fossero divisibili in modo lineare è possibile applicare una trasfomazione non lineare ai dati, per portarli in uno spazio con più dimensioni nel quale sono divisibili in modo lineare, la funzione che effettua questa trasformazione non lineare è detta **kernel** e può essere scelta in base al problema specifico, possibili scelte sono il kernel polinomiale o la radial basis function (RBF). Inoltre non sempre i sample sono divisibili perfettamente ma è necessario commettere degli errori, in questo caso si utilizza il **coefficiente di regolarizzazione** (C), un iperparametro per gestire il bilanciamento tra errori commessi e la grandezza del margine: per C piccolo si dà più importanza al margine, mentre per C grande si cerca di evitare più errori possibile. L'SVM ha il vantaggio di occupare poca memoria ed essere molto efficace in caso di dimensionalità delle features molto elevata, grazie all'utilizzo dei kernel.

L'ultimo metodo di learning approfondito sono le **reti neurali**. Una rete è composta da diversi *layer*, e ogni layer ha diversi *nodi*. Le features sono messe in input al primo layer, che a sua volta calcolerà una combinazione lineare degli ingressi e poi ne farà una trasformazione non

lineare secondo una certa *funzione di attivazione*, dopodiché l'output del primo layer è mandato in input al secondo layer. Questo processo è iterato fino ad arrivare all'ultimo layer, che calcolerà l'output finale. Il numero di layer e il numero di nodi per ogni layer è un iperparametro da scegliere opportunamente, inoltre è possibile scegliere la funzione di attivazione da applicare ai dati, in particolare le funzioni cosiddette "S-shaped", come la funzione sigmoide e la funzione tangente iperbolica (tanh) caratterizzano le reti chiamate *multilayer perceptron* (MLP). Un'altra componente fondamentale delle reti neurali sono i pesi che ogni collegamento tra nodi di livelli diversi ha, questi pesi sono i valori che il modello deve imparare durante la fase di training, per farlo le reti neurali usano il **error backporpagation**: si confronta il valore in uscita con il valore reale atteso, e in base all'errore, cioè alla differenza tra i due, si modificano i pesi. Un alto numero di layer e di nodi può portare a overfitting e ad alta complessità, per evitare questo problema di utilizzano delle tecniche di regolarizzazione, come il **weight decay** e l'**early stopping**. Un difetto delle reti neurali è la loro lentezza nel training dovuta proprio all'applicazione dell'algoritmo di error backpropagation, infatti le reti neurali richiedono molte risorse computazionali e l'addestramento risulta quindi lento.

Per questo progetto è stato scelto di utilizzare tre dei modelli descritti:

- Decision Tree,
- Support Vector Machine
- Reti Neurali

In particolare per ogni modello saranno studiate tre versioni:

- versione 1: i modelli sono addestrati con i dati grezzi e originali, senza pre-processing,
- versione 2: i modelli sono addestrati con i dati pre-processati,
- versione 3: si utilizza una versione ensemble della versione migliore tra le prime due

4 Dettagli implementativi

In questo paragrafo verranno descritti in modo specifico i dettagli implementativi di ogni modello e di ogni sua versione, oltre ai vari passaggi di pre-processing e il metodo di valutazione.

4.1 Pre-processing

Per le versioni 1 dei modelli i dati non vengono pre-processati, infatti i dati non vengono pre-processati come descritto nella sezione 2, ma vengono solamente fatte le modifiche in fase di EDA, modificando quindi i tipi dei dati ed eliminando i valori NaN. Lo splitting in questo caso è fatto quindi sui dati originali, togliendo dalle features lo *user*, che è il nome del soggetto e non influisce sull'apprendimento, ed è stato tolta anche la feature *gender*, in quanto non ancora trasformata con l'encoder. Inoltre la variabile target *class* in questo caso rimane sottoforma di stringa.

Per le versioni 2 invece vengono effettuati gli step di pre-processing descritti brevemente nella sezione 2, in particolare dopo aver fatto l'encoding della feature *gender* e del target *class*, è stata studiata la matrice di correlazione, mostrata in figura 7, per valutare la collinearità tra features tra di loro, e con la variabile target. Ne risulta che le variabili *y1* e *y4* sono particolarmente correlate con la variabile target, quindi saranno importanti ai fini della classificazione. In secondo luogo è possibile notare che le features *how_tall_in_meters*, *weight* e *body_mass_index* sono particolarmente correlate tra di loro e di conseguenza è conveniente eliminarne due mantenerne una sola. Infine le tre variabili che rappresentano le coordinate dell'accelerometro 2, quello sulla coscia sinistra, sono molto correlate tra loro, quindi anche in questo caso conviene mantenerne una sola.

Dalle informazioni raccolte dalla matrice di correlazione, in figura 7, effettuiamo la features selection eliminando le variabili *weight* e *body_mass_index* perchè altamente correlate con *how_tall_in_meters* e viene mantenuta quest'ultima; vengono anche eliminate le coordinate x e z

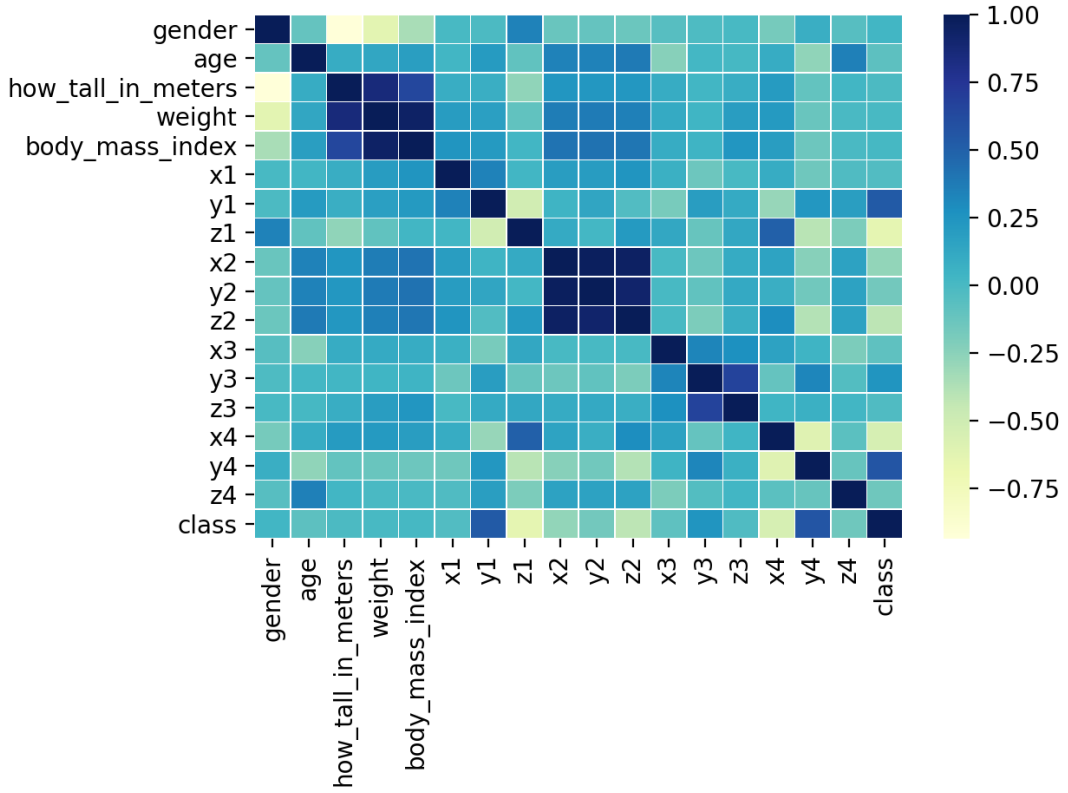


Figure 7: Matrice di correlazione.

del secondo accelerometro, mantenendo solo la coordinata y. Infine vengono eliminate le variabili *user* e *age* in quanto non influenti sul riconoscimento della postura.

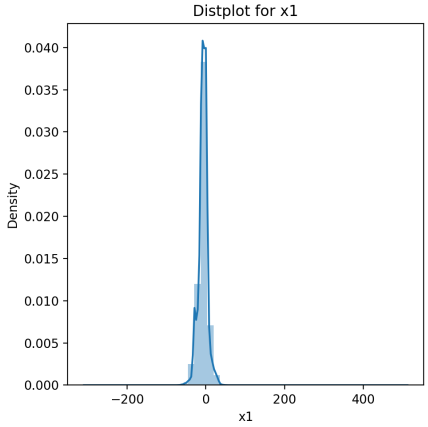
Le features selezionate vengono poi sottoposte a uno scaling, in particolare di applica il metodo del **min-max-scaling**: trasforma il valore di ogni features portandolo a un valore compreso tra 0 e 1, per farlo applica a ogni variabile la seguente formula:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

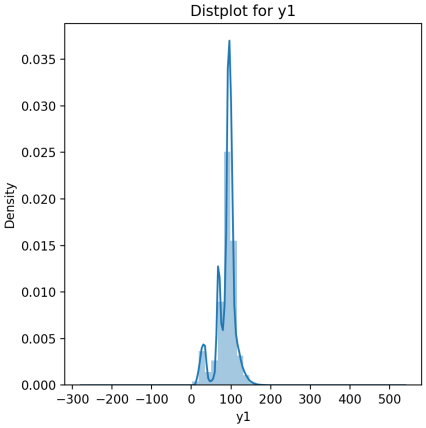
È importante sottolineare che i parametri $min(x)$ e $max(x)$ vanno calcolati sulla base del solo training set, dopodiché si applica la trasformazione al training set e al test set.

In figura 8 sono mostrati i grafici di distribuzione di alcune variabili prima di aver subito il min-max-scaling, come si vede i valori variano nell'ordine delle centinaia intorno allo zero.

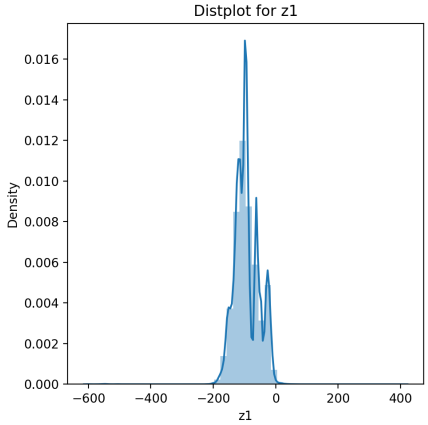
In figura 9 invece sono mostrati i grafici delle distribuzioni delle stesse variabili dopo aver subito la trasformazione del min-max-scaling, come si vede il valore è compreso tra 0 e 1 ma l'andamento è lo stesso e la distribuzione mantiene la stessa forma, a prova del fatto che i dati mantengono le stesse informazioni rappresentative. Grazie a quest'operazione di trasformazione delle features i modelli saranno più efficienti in fase di training e veloci.



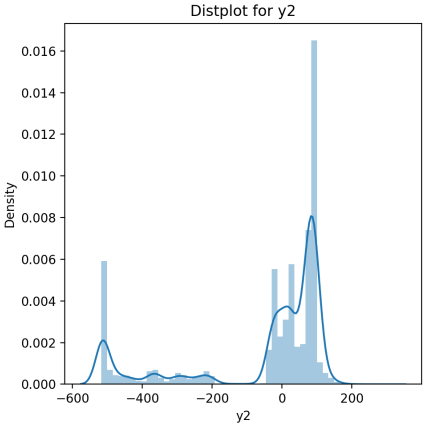
(a) Distribuzione di x1 prima del min-max-scaling.



(b) Distribuzione di y1 prima del min-max-scaling.



(c) Distribuzione di z1 prima del min-max-scaling.



(d) Distribuzione di y2 prima del min-max-scaling.

Figure 8: Distribuzione di variabili prima del min-max-scaling.

4.2 Decision Tree

4.3 Support Vector Machine

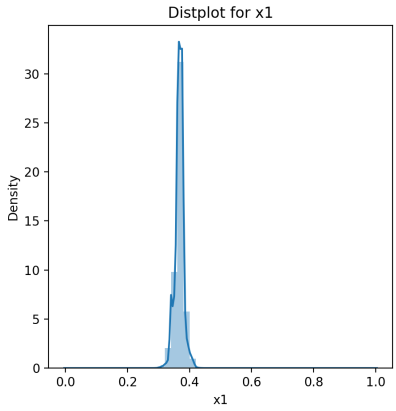
4.4 Reti neurali

4.5 Metriche di valutazione

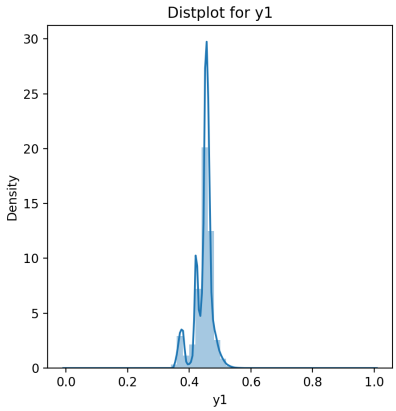
5 Risultati e discussione

6 LaTeX Examples

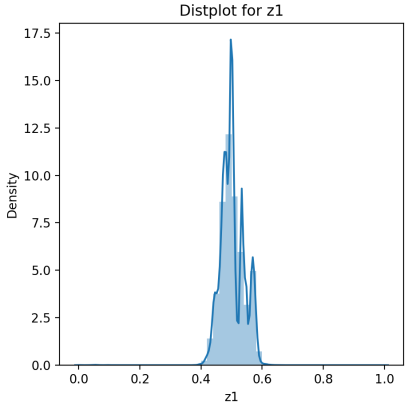
The following examples should help you to write your technical report using LaTeX. You'll find here the examples of tables, figures, citations and references. For other features of LaTeX, see tutorials on **Overleaf** or use this **cheatsheet**. To work with this template, download its entire folder (including /sections, /bibliography and /figures), and run your LaTeX editor like **Overleaf**.



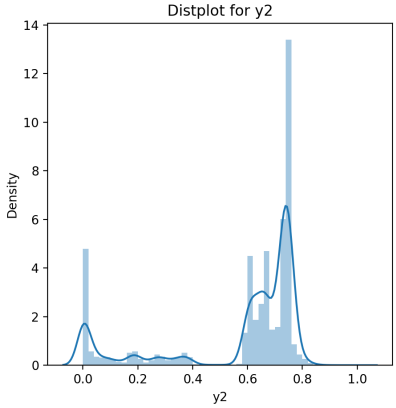
(a) Distribuzione di x1 dopo il min-max-scaling.



(b) Distribuzione di y1 dopo il min-max-scaling.



(c) Distribuzione di z1 dopo il min-max-scaling.



(d) Distribuzione di y2 dopo il min-max-scaling.

Figure 9: Distribuzione di variabili dopo il min-max-scaling.

Example Citation

Example of citation: [**?**] and [**?**].

Example References

Example of table reference: see Table 1.
Example of equation reference: see Equation (2).
Example of reference to Section **??**.
Example of reference to Subsection **??**.
Example of figure reference: see Figure 10.
Example of subfigure reference: see Figure 11a.

Example list

- Bullet point one
- Bullet point two
- Nested list items:

- Nested item one
- Nested item two

Enumerations

1. Numbered list item one
2. Numbered list item two
3. Nested list items:
   (a) Nested item one
   (b) Nested item two

Example Table

Treatments	**Response 1**	**Response 2**
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Table 1: Table caption

Example Equation

Equations within the text: $e = mc^2$. Equation with label on its own line:

$$e = mc^2 \tag{2}$$

Example Figures

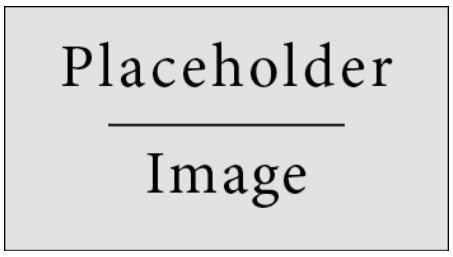
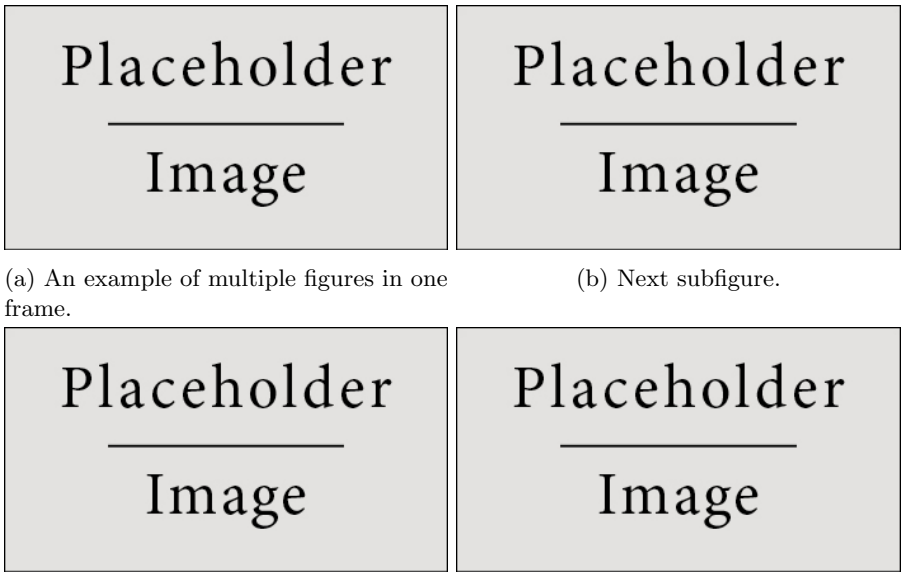


Figure 10: An example of simple figure.



(a) An example of multiple figures in one frame.

(b) Next subfigure.

(c) Subfigure on another line.

(d) Yet another subfigure.

Figure 11: More figures in appendix.