# 7. Data Profiling

Successivamente alla creazione del modello si è provveduto ad effettuare diverse operazioni di data profiling, come l'estrazione di dipendenze funzionali, dipendenze funzionali rilassate [9] e features selection. Tecniche di tale genere aiutano nella scoperta, comprensione e organizzazione dei dati. Operazione preliminare di tali processi è stata la creazione di due differenti csv. Questi sono stati utilizzati come input per i vari algoritmi di FD discovery e features selection.

Il primo file csv riportava tutte le features del dataset, cioè:

* Image\_Name
* Glasses
* Genders
* Reflections
* Image\_quality
* Sensor\_type
* Eye\_state

Il secondo file csv comprendeva tutte le features del dataset tranne l'attributo " Eye\_state", che rappresenta la feature target del dataset.

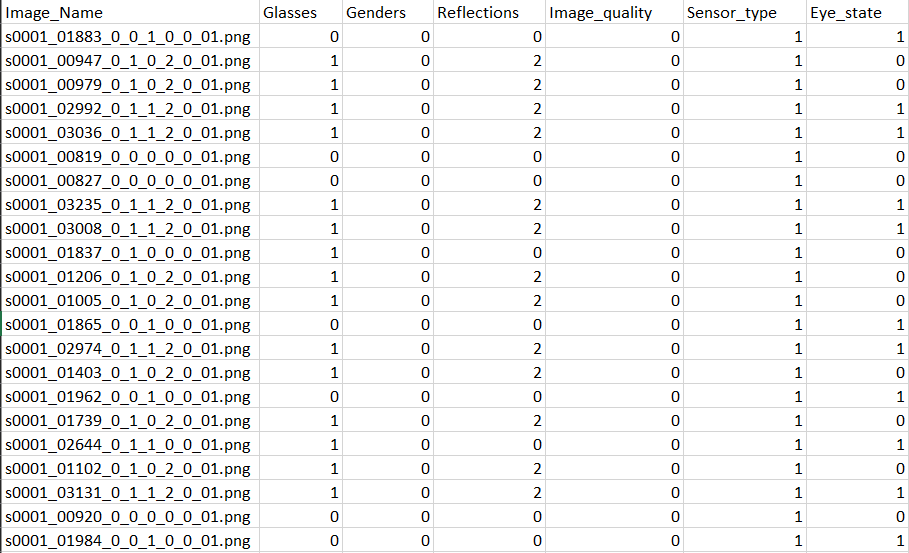


Figura 7 File CSV con etichette

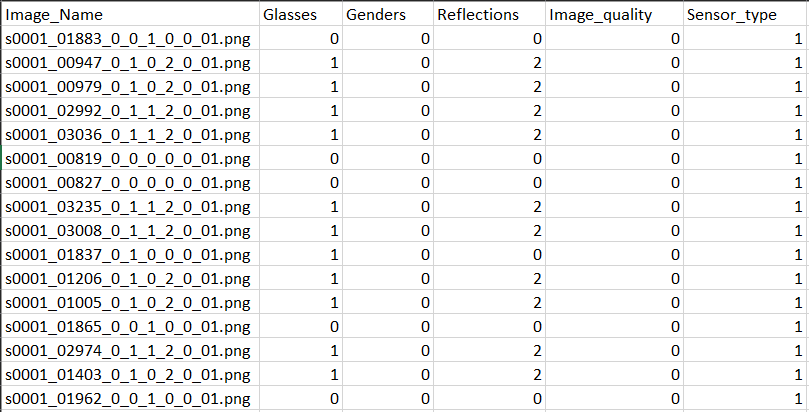


Figura 18 File CSV senza etichette

## 7.1 FD

La prima operazione è stata quella di ricercare l'esistenza di dipendenze funzionali all'interno del dataset. Tale operazione è stata effettuata tramite il tool COD3. Il file di input utilizzato per l'algoritmo è stato quello che non prevedeva il target attribute. In aggiunta, il file è stato modificato in modo da eliminare l'attributo Image\_Name, non rilevante allo scopo.

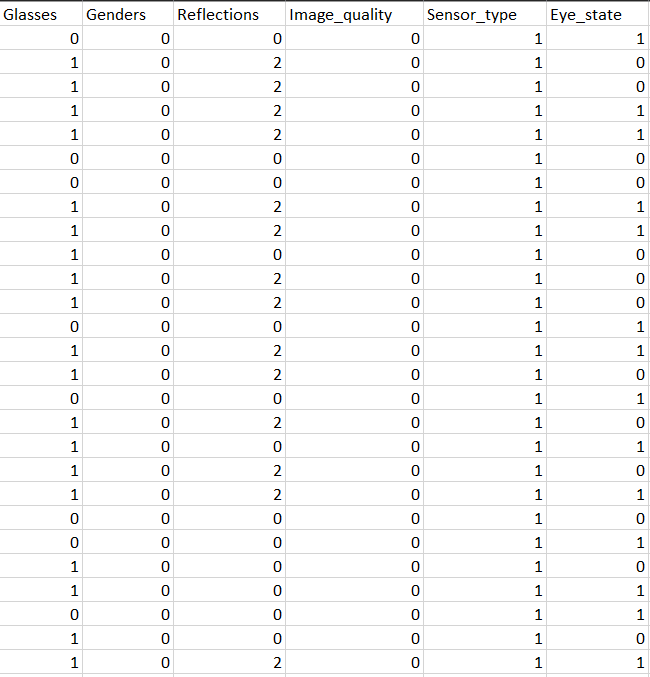


Figura 19 Struttura del file input per l'algoritmo COD3

Applicando l'algoritmo per l'FD discovery su tale file non sono state rilevate dipendenze funzionali.

Tuttavia, esaminando i log del tool, si è notato che inizialmente erano state identificate quattro differenti dipendenze, che diventano non più valide oltre un certo numero di tupla.

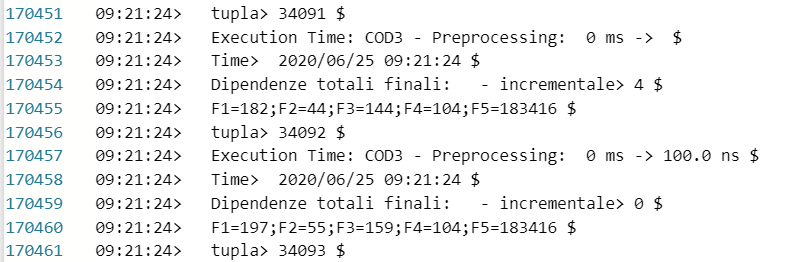


Figura 19 Algoritmo COD3: Punto di terminazione di validità delle FD

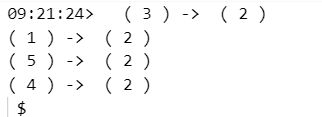


Figura 20 Algoritmo COD3: Dipendenze rilevate fino alla tupla 34091

Nello specifico, non avendo considerato l'attributo Image\_Name, la corrispondenza degli attributi era la seguente:

|  |  |
| --- | --- |
| Nome | Numero |
| Glasses | 1 |
| Genders | 2 |
| Reflections | 3 |
| Image\_quality | 4 |
| Sensor\_type | 5 |

Tabella Mapping delle features

Per cui, le corrispondenze trovate erano:

Glasses →Genders

Reflections → Genders

Image\_quality → Genders

Sensor\_type → Genders

Si è quindi provveduto ad analizzare il file allo scopo di trovare il motivo della terminazione della validità della dipendenza.

Immagine che contiene mappa

Descrizione generata automaticamente

Figura 1 Dettaglio del file csv in cui si interrompe la validità delle dipendenze

Come si può analizzare dalla Figura 21, nella tupla 34092 si ha la prima occorrenza del valore 1 per l'attributo "Genere". Ciò vuol dire che dalla tupla 1 alla tupla 34091 i valori di "Genere" erano tutti uguali, e pari a 0. Il cambiamento del valore di tale attributo ha provocato la terminazione della validità delle dipendenze. Le dipendenze rilevate sono state considerate banali e non utili ai fini della feature selection, in quanto basate sulla considerazione di un valore univoco sulla parte destra.

## 7.2 RFD

Recentemente si è posto un interesse sulle cosiddette Relaxed Functional Dependencies [9], vale a dire delle dipendenze funzionali inesatte che rilassano su uno o più vincoli delle dipendenze funzionali canoniche. Non essendo state rilevate dipendenze funzionali canoniche sul dataset, si è provveduto a ricercare RFD che rilassavano sull'extent e sul confronto. A tal fine sono stati utilizzati gli algoritmi BIRD e DOMINO [10] per trovare le RFD che rilassavano sull' extent, con quest'ultimo utilizzato anche per rilevare le RFD che rilassavano sul comparison.

### **7.2.1 Extent**

Rilassare sull'extent significa che la dipendenza funzionale vale per "quasi tutte" le tuple, o su un sottoinsieme di esse. Diversi sono i casi per cui una dipendenza potrebbe non valere per tutte le tuple, come la presenza di valori mancanti, differenti tipi di dati o errori presenti negli stessi.

Applicando l'algoritmo sul dataset considerato, è stata rilevata la seguente dipendenza funzionale approssimata:

(Genders Reflections Sensor\_type) -> (Glasses)

con soglia di extent g3-error di 0.9.

Il g3-error corrisponde al minimo numero di tuple da rimuovere da una istanza di relazione r affinché una dipendenza funzionale X → Y valga. Nello specifico caso, ciò significa che, affinchè la dipendenza trovata valga, sarebbe necessario eliminare il 90% delle tuple. Nonostante la soglia della dipendenza sia molto alta, si è effettuata la costruzione di un nuovo modello che considerasse tutte le features tranne la feature Glasses, che rappresenta la parte destra della dipendenza funzionale, quindi la feature che è possibile derivare dalle altre. La struttura del modello, a meno delle features utilizzate per il training, è la medesima del modello descritto nella sezione 5.

In Figura 22 sono mostrati i grafici riportanti l'andamento della loss e dell'accuracy al crescere del numero di epoche:

Immagine che contiene testo, mappa

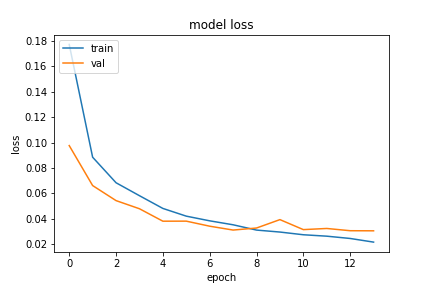
Descrizione generata automaticamente

Figura 22 Andamento di accuracy e loss del modello descritto in 7.2.1

I risultati del modello sono mostrati nella tabella 11:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Training** | | **Validation** | | **Test** | |
| Nome | **Mean epoch training time** | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| all-glasses\_14\_0.25\_  3CL\_0.2\_  basicModel\_24 | 10.926 | 0.051 | 0.980 | 0.043 | 0.984 | 0.032 | 0.989 |

Tabella Risultati modello descritto in 7.2.1

Di seguito viene mostrato il confronto tra il modello discusso nel 6. e il modello discusso in questo paragrafo.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Training** | | **Validation** | | **Test** | |
| Nome | **Mean epoch training time** | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| all-glasses\_14\_0.25\_  3CL\_0.2\_  basicModel\_24 | 10.926 | 0.051 | 0.980 | 0.043 | 0.984 | 0.032 | 0.989 |
| all\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 10.772 | 0.057 | 0.978 | 0.044 | 0.984 | 0.041 | 0.985 |
|  | + 0.154 | - 0.06 | + 0.002 | -0.001 | 0 | - 0.009 | + 0.004 |

Tabella Confronto tra i modelli di 5. e 7.2.1

Come si può notare dalla tabella, i risultati ottenuti non variano di quantità rilevanti.

### **7.2.2 Confronto**

Rilassare le dipendenze funzionali sul confronto significa usare paradigmi di matching approssimato per confrontare i valori dell'attributo sul lato sinistro e quelli del lato destro della dipendenza funzionale. Ciò permette di catturare relazioni semantiche tra gruppi di valori che sembrano essere simili, piuttosto che identici. Generalmente, questo tipo di RFD viene utilizzato per risolvere problemi di data quality, query optimization e knowledge discovery. Queste si basano su una soglia di comparison, che è la differenza massima ammissibile tra i valori di due tuple su un attributo.

Tramite l'algoritmo DOMINO, si è provato ad estrarre RFD basate sul confronto dal dataset utilizzato nel progetto. Nel fare ciò si sono utilizzate differenti soglie di comparison: 1, 2, 4 e 15.

Utilizzando una soglia di comparison pari ad uno non sono stati trovati risultati, vale a dire che non vi sono due attributi X e Y per cui la differenza massima tra i valori di X sia minore ad uno e la differenza massima tra i valori di Y sia minore ad 1.

Di seguito vengono riportati i risultati ottenuti con le diverse soglie:

SOGLIA 2

\*\*\*\*\*\* DISCOVERED RFDs \*\*\*\*\*\*\*

0@2.0->0@2.0 cc:0.9685232103301251

1@2.0->0@2.0 cc:0.8694514821618544

0@2.0->1@2.0 cc:0.5990717281682707

SOGLIA 4

\*\*\*\*\*\* DISCOVERED RFDs \*\*\*\*\*\*\*

0@4.0->0@2.0 cc:0.9685232103301251

1@4.0->0@2.0 cc:0.8694514821618544

0@4.0->1@2.0 cc:0.5990717281682707

SOGLIA 15

\*\*\*\*\*\* DISCOVERED RFDs \*\*\*\*\*\*\*

0@15.0->0@2.0 cc:0.9685232103301251

1@15.0->0@2.0 cc:0.8694514821618544

0@15.0->1@2.0 cc:0.5990717281682707

Nonostante le tre diverse soglie, le RFD sul confronto e le cc [[1]](#footnote-1) (le soglie delle RFD minimali) rilevate dal tool sono le medesime:

0 → 0

1 → 0

0→ 1

E cioè:

Glasses → Glasses

Genders → Glasses

Glasses → Genders

Poichè la prima dipendenza è banale, ed il modello senza la feature Glasses è stato descritto nell' 7.2.1, si discuterà del modello sviluppato includendo tutte le features tranne Genders. La struttura del modello, a meno delle features utilizzate per il training, è la medesima del modello descritto nel 6.

In Figura 23 sono mostrati i grafici riportanti l'andamento della loss e dell'accuracy al crescere del numero di epoche:

Immagine che contiene mappa, testo

Descrizione generata automaticamenteImmagine che contiene screenshot

Descrizione generata automaticamente

Figura 23 Andamento di accuracy e loss del modello descritto in 7.2.2

I risultati del modello sono mostrati nella Tabella 13:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Training** | | **Validation** | | **Test** | |
| Nome | **Mean epoch training time** | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| all-genders\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 10.995 | 0.050 | 0.981 | 0.042 | 0.985 | 0.040 | 0.987 |

Tabella Risultati modello descritto in 7.2.2

Di seguito viene mostrato il confronto tra il modello discusso nel 6. e il modello discusso in questo paragrafo.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Training** | | **Validation** | | **Test** | |
| Nome | **Mean epoch training time** | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| all-genders\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 10.995 | 0.050 | 0.981 | 0.042 | 0.985 | 0.040 | 0.987 |
| all\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 10.772 | 0.057 | 0.978 | 0.044 | 0.984 | 0.041 | 0.985 |
|  | + 0.223 | -0.007 | + 0.003 | -0.002 | + 0.001 | -0.001 | + 0.002 |

Tabella Confronto tra i modelli di 5. e 7.2.2

Come si può notare dalla Tabella 14, i risultati ottenuti non variano di quantità rilevanti.

## 7.3 Feature selection

La feature selection è un importante concetto del machine learning, che può influire in maniera rilevante sulle performance di un modello. Infatti, features irrilevanti possono penalizzare le performance dello stesso, per cui si rende necessaria una attenta fase di scelta delle feature da utilizzare. La feature selection può essere intesa come un processo in cui vengono selezionate (automaticamente o manualmente) le feature più rilevanti per la feature target, cioè la variabile predittiva.

In generale, una operazione di questo tipo consente di:

1. Ridurre l'Overfitting:

Meno features corrispondono a una minore probabilità di effettuare decisioni basate su noisy data.

1. Migliorare l'Accuracy

In questo progetto sono state utilizzate tre tecniche tra le diverse presenti in letteratura. Esse sono:

1. Selezione Univariata
2. Feature Importance
3. Matrice di Correlazione con Heatmap

Nel seguito verranno mostrate le diverse tecniche applicate sul dataset utilizzato nel progetto, con degli snippet di codice per mostrarne il funzionamento.

Operazione preliminare alle tecniche di feature selection è stata quella di separare la colonna target dalle altre. Ciò è stato effettuato tramite il codice di Figura 24.

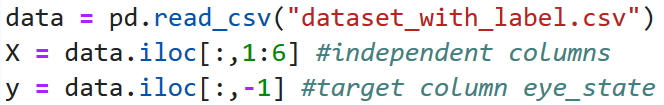


Figura 24 Separazione attributo target

### **7.3.1 Selezione Univariata**

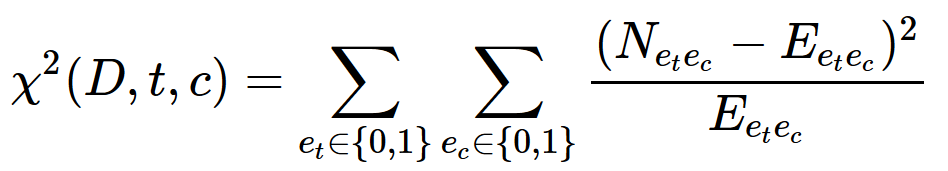
I test statistici sono uno dei modi utilizzati per selezionare le features che hanno la più alta correlazione con la variabile di output.

La selezione univariata seleziona le migliori features basati su test statistici univariati. Viene confrontata ogni feature alla variabile target, per esaminare se vi è una relazione statisticamente rilevante fra loro. Analizzando la relazione tra una feature e la variabile target, le altre features vengono ignorate. Per tale motivo tale tecnica viene chiamata univariata. Da ciò ne deriva che ogni feature ha un proprio punteggio. Alla fine del processo, i punteggi delle features vengono confrontati, selezionando solo quelli più alti.

Uno dei test statistici che è possibile utilizzare è quello del chi-squared ().

Il test del è usato in statistica per verificare l'indipendenza di due eventi. Nella feature selection, invece, viene utilizzato per testare se l'occorrenza di uno specifico termine e l'occorrenza di una specifica classe sono indipendenti.

Formalmente, dato un documento , stimiamo la seguente quantità per ogni termine e li ordiniamo per punteggio:



Dove:

* è la frequenza osservata ed rappresenta la frequenza attesa
* è 1 se il documento contiene il termine , altrimenti
* è 1 se il documento è nella classe , altrimenti

Per ogni feature (termine), un alto punteggio nel test indica che l'ipotesi nulla di indipendenza

deve essere rifiutata, e l'occorrenza del termine e della classe sono dipendenti. In questo caso la feature deve essere considerata.

La classe SelectKBest della libreria di Scikit-learn consente di effettuare un insieme di test statistici per selezionare uno specifico numero k di features. Tra questi è presente anche il test del .

Di seguito viene mostrato il codice per restituire le 5 features del dataset insieme ai relativi punteggi, utilizzando il test .

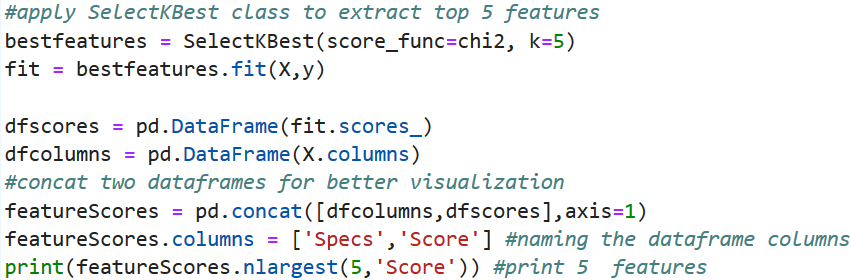


Figura 25 Snippet per la selezione univariata.

Il risultato di tali operazioni è mostrato nella Figura 26.

Immagine che contiene uccello

Descrizione generata automaticamente

Figura 26 Risultati selezione univariata.

Analizzando la Figura 26, si può notare che le features che sembrano essere più correlate all' attributo target risultano essere nell'ordine "Reflections","Sensor\_type" e "Glasses".

Si è creato un nuovo modello che considerasse tali features, oltre alle immagini. La struttura del modello, a meno delle features utilizzate per il training, è la medesima del modello descritto nella sezione 5.

In Figura 27 sono mostrati i grafici riportanti l'andamento della loss e dell'accuracy al crescere del numero di epoche:

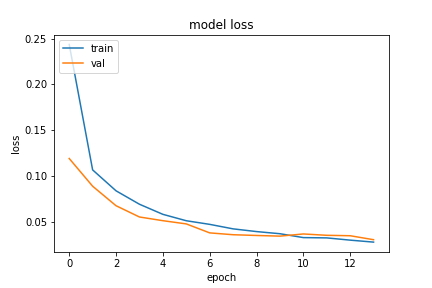
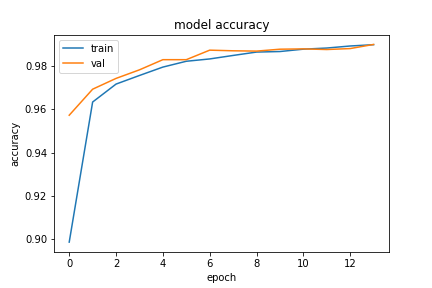


Figura 27 Andamento di accuracy e loss del modello descritto in 7.3.1

I risultati del modello sono mostrati nella Tabella 15:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Training** | | **Validation** | | **Test** | |
| Nome | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| img,reflections,sensorType,glasses\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 0.064 | 0.976 | 0.050 | 0.982 | 0.035 | 0.988 |

Tabella 15 Risultati modello descritto in 7.3.1

Di seguito viene mostrato il confronto tra il modello discusso nella sezione 5. e il modello discusso in questo paragrafo.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Training** | | **Validation** | | **Test** | |
| Nome | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| img,reflections,sensorType,glasses\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 0.064 | 0.976 | 0.05 | 0.982 | 0.035 | 0.988 |
| all\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 0.057 | 0.978 | 0.044 | 0.984 | 0.041 | 0.985 |
|  | + 0.007 | - 0.002 | + 0.006 | - 0.002 | - 0.006 | + 0.003 |

Tabella 16 Confronto tra i modelli di 5. e 7.3.1

Come si può notare dalla Tabella 16, i risultati ottenuti non variano di quantità rilevanti.

### **7.3.2 Feature Importance**

Alternativamente alla selezione univariata, si può considerare l'importanza di ciascuna feature del dataset utilizzando una apposita proprietà del modello, cioè quella della "feature importance". Questa restituisce un punteggio per ogni feature del dataset: più è alto il punteggio e più la feature è rilevante rispetto alla feature target.

Tale proprietà è una classe propria dei classificatori ad alberi. Per usarla, quindi, ci sarà la necessità di usare un'istanza di questo tipo di classificatori.

In particolare, viene utilizzato il cosiddetto Extremely Randomized Trees Classifier o più semplicemente Extra Trees Classifier, che è una tecnica di ensemble learning che aggrega i risultati di diversi decision trees non correlati raccolti in una foresta, per poi restituire il risultato di classificazione di quest'ultima. Esso differisce da un tradizionale classificatore Random Forest nel modo di costruire i decision trees nella foresta. Ogni Decision Tree nella Extra Trees Forest è costruito a partire dai dati di training originali. Successivamente, per ogni nodo, ad ogni albero viene passato in input un campione casuale delle k features dall’insieme delle feature, da cui ogni decision tree deve selezionare le migliori feature per dividere i dati basandosi su un particolare criterio (spesso si utilizza l’indice Gini). Tale campione casuale delle features porta alla creazione di diversi decision tree non correlati.

con

= probabilità che un oggetto venga classificato in una determinata classe

Per eseguire l’operazione di feature selection utilizzando la struttura della foresta descritta, durante la costruzione della foresta, per ogni feature, viene calcolata la riduzione normalizzata totale nel criterio matematico utilizzato nella decisione dello split delle feature. Considerando come criterio matematico l’indice Gini, la riduzione normalizzata totale viene chiamata Gini Importance. Per effettuare la feature selection, ogni feature è ordinata per importanza in modo discendente, a seconda della propria Gini Importance. Successivamente, si considerano le prime k feature, a seconda del k scelto dall’utente.

Nello snippet di codice seguente viene mostrato come sia possibile estrarre i punteggi per le 5 feature del dataset. Il criterio utilizzato di default dall' ExtraTreesClassifier è l'indice Gini.

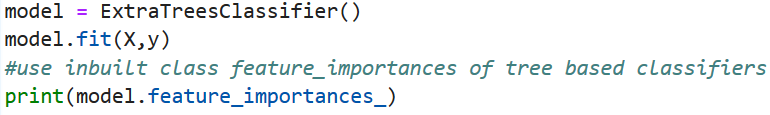


Figura 28 Costruzione del classificatore ad albero

Stampando il contenuto della variabile *model.feature\_importances\_* si ottiene una lista di valori indicante l'importanza di ognuna delle feature del dataset.



Figura 29 Risultati della feature importance

L'ordine dei risultati restituiti si riferisce a quello del salvataggio degli attributi nel dataframe. Per permettere una visualizzazione più chiara è possibile creare un grafico che mostri, oltre al valore delle feature, anche la loro denominazione.

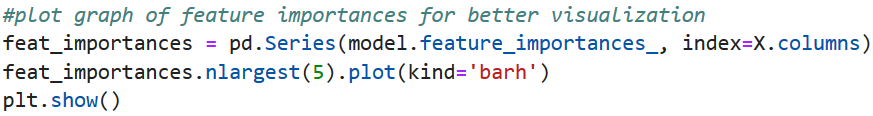


Figura 30 Snippet per la costruzione dell'istogramma della feature importance

Eseguendo il codice si ottiene il seguente grafico:

Immagine che contiene screenshot

Descrizione generata automaticamente

Figura 31 Istogramma della feature importance

Dal grafico in Figura 31, è possibile osservare che la feature che ha riportato il maggiore punteggio è stata "Sensor\_type", differentemente a quanto accaduto con il metodo della selezione univariata descritto nel 7.3.1. Le altre features del dataset presentano dei punteggi di importanza molto minori di quello ottenuto da "Sensor\_type", addirittura inferiori alla metà di quest'ultimo.

Si è quindi creato un nuovo modello che considerasse la feature "Sensor\_type" oltre alle immagini. La struttura del modello, a meno delle features utilizzate per il training, è la medesima del modello descritto nella sezione 5.

In Figura 32 sono mostrati i grafici riportanti l'andamento della loss e dell'accuracy al crescere del numero di epoche:

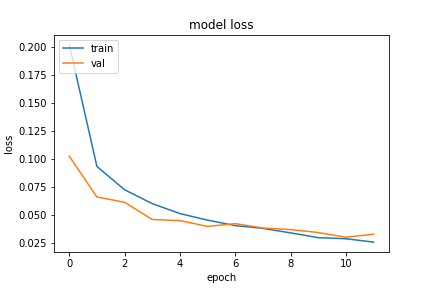
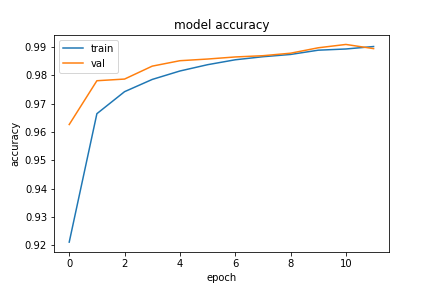


Figura 32 Andamento di accuracy e loss del modello descritto in 7.3.2

I risultati del modello sono mostrati nella Tabella 15:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Training** | | **Validation** | | **Test** | |
| Nome | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| img,sensorType\_12\_0.25\_3CL\_0.2\_basicModel\_24 | 0.059 | 0.977 | 0.047 | 0.983 | 0.039 | 0.987 |

Tabella 17 Risultati modello descritto in 7.3.2

Di seguito viene mostrato il confronto tra il modello discusso nella sezione 5. e il modello discusso in questo paragrafo.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Training** | | **Validation** | | **Test** | |
| Nome | **Loss** | **Accuracy** | **Loss** | **Accuracy** | **Loss** | **Accuracy** |
| img,sensorType\_12\_0.25\_3CL\_0.2\_basicModel\_24 | 0.059 | 0.977 | 0.047 | 0.983 | 0.039 | 0.987 |
| all\_14\_0.25\_3CL\_0.2\_basicModel\_24 | 0.057 | 0.978 | 0.044 | 0.984 | 0.041 | 0.985 |
|  | + 0.002 | - 0.001 | + 0.003 | - 0.001 | -0.002 | + 0.002 |

Tabella 18 Confronto tra i modelli di 5. e 7.3.2

Come si può notare dalla Tabella 18, i risultati ottenuti non variano di quantità rilevanti.

### **7.3.3 Matrice di Correlazione con Heatmap**

Un ulteriore metodo per la feature selection è rappresentato dall'analisi della matrice di correlazione delle features. Essa mostra come le features sono correlate rispetto alle altre o all'attributo target. La correlazione può essere positiva o negativa. Nel primo caso l'incremento del valore della feature provoca l'incremento di quello dell'attributo target, mentre nel secondo l'incremento del valore della feature provoca il decremento del valore dell'attributo target. Il modo più comune di calcolare la correlazione è tramite l'utilizzo del coefficiente di Pearson. Esso misura soltanto le relazioni lineari tra due variabili, non riuscendo a rilevare relazioni non-lineari. Il valore del coefficiente di Pearson varia da -1 a +1, dove:

* + 1 descrive una correlazione lineare positiva perfetta
* - 1 descrive una correlazione lineare negativa perfetta
* 0 indica che non vi è nessuna correlazione

Una matrice di correlazione, quindi, non è altro che l'insieme dei valori di correlazione tra tutte le coppie di attributi. Essa è simmetrica e presenta tutti gli elementi diagonali pari ad 1, essendo 1 il valore della correlazione di un elemento con sé stesso.

Per una migliore lettura della stessa, la matrice di correlazione può essere associata ad una heatmap. Questa è una tecnica di data visualization che mostra la magnitudine di un fenomeno attraverso un colore.

Tramite la heatmap è facile verificare quali features sono più correlate alla variabile target. Il seguente codice mostra come sia possibile creare una matrice di correlazione associata ad una heatmap, ottenuta tramite la libreria Seaborn.

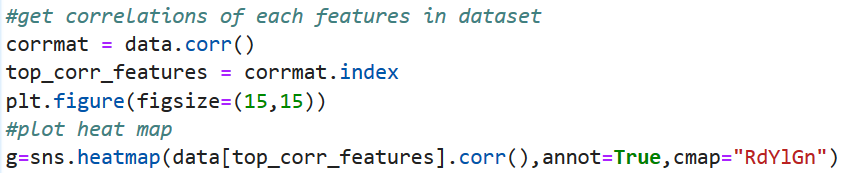


Figura 33 Snippet per la costruzione della matrice di correlazione con heatmap

Eseguendo le operazioni, si ottiene il risultato in Figura 34.

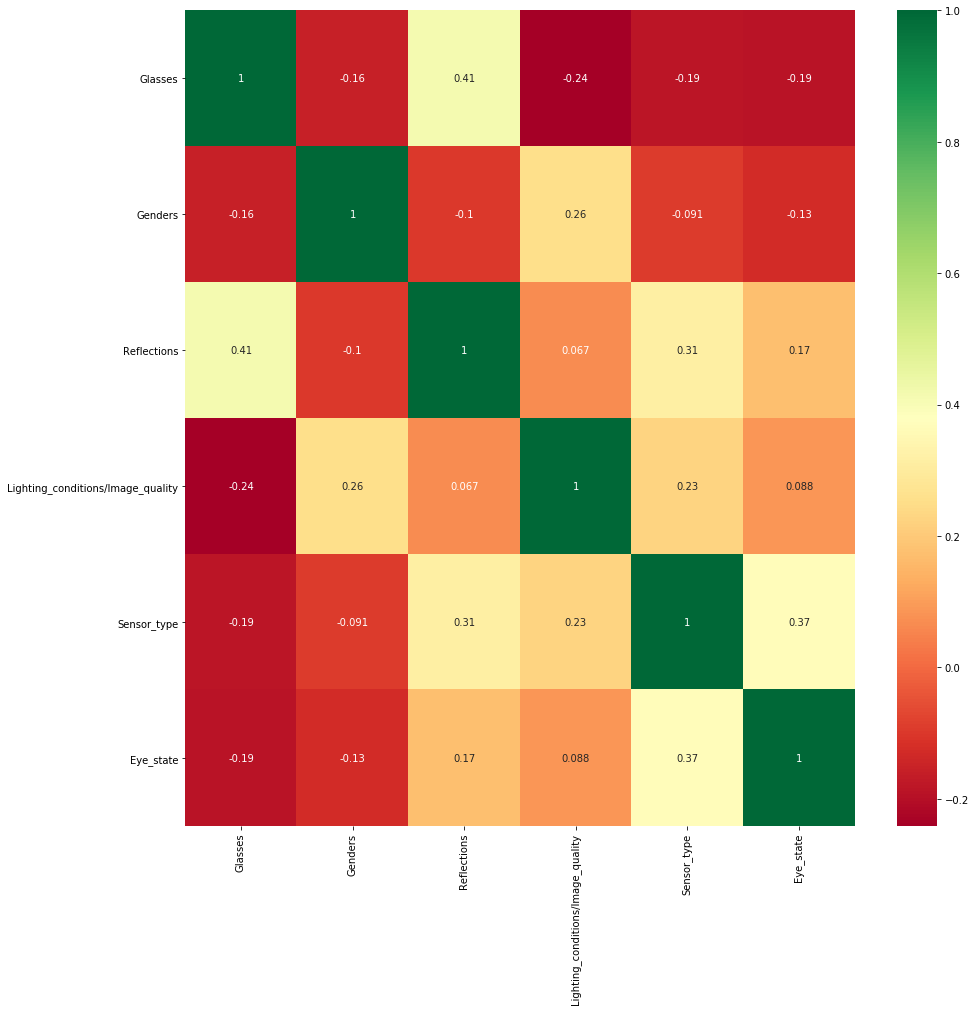


Figura 34 Matrice di correlazione con heatmap

Analizzando l'immagine in Figura 34, si nota che nessuna feature è fortemente correlata con la variabile target "Eye\_state": la correlazione più alta è risultata essere 0.37, corrispondente alla feature "Sensor\_type".

Nello specifico caso in questione, l'applicazione dei risultati ottenuti da tale tecnica di feature selection non migliora le performance del modello. Infatti, come si è visto dall'analisi della tecnica di Feature Importance nella sezione 7.3.2, il modello che considera solo immagini e la feature Sensor\_type non ottiene risultati molto diversi dal modello utilizzato nella sezione 5.

1. Per i dettagli dell'algoritmo DOMINO, si faccia riferimento alla sezione 3.4 del [10] [↑](#footnote-ref-1)