

# Analisi di metodi di machine learning per la classificazione di immagini di prodotti biodegradabili

Laureando  
Bruno Francesco Nocera

Relatore  
Prof. Irene Amerini



**SAPIENZA**  
UNIVERSITÀ DI ROMA



**SAPIENZA**  
UNIVERSITÀ DI ROMA

# Analisi di metodi di machine learning per la classificazione di immagini di prodotti biodegradabili

**Facoltà di Ingegneria dell'informazione, informatica e statistica**  
**Dipartimento di Informatica**  
**Corso di laurea in Informatica**

**Bruno Francesco Nocera**  
**Matricola 1863075**

Responsabile  
Prof. Irene Amerini

A handwritten signature in blue ink, likely belonging to Prof. Irene Amerini.

Corresponsabile  
Lorenzo Papa

A.A. 2021-2022

# INDICE

<b>Abstract</b> .....	<b>3</b>
<b>1. Introduzione</b> .....	<b>4</b>
<b>2. Algoritmi di Machine Learning</b> .....	<b>7</b>
2.1 Support Vector Machines (SVM).....	8
2.2 K- Nearest Neighbor (KNN) .....	12
2.3 Random Forest .....	14
<b>3. Soluzioni proposte</b> .....	<b>17</b>
3.1 Feature selection .....	17
3.2 Edge detection.....	17
3.3 Discrete Fourier Transform (DFT).....	19
3.4 Entropia .....	21
3.5 Analisi spazio RGB.....	22
3.6 Principal Component Analysis (PCA).....	22
<b>4. Risultati</b> .....	<b>25</b>
4.1 Dataset .....	25
4.2 Support Vector Machine (SVM) .....	27
4.3 K- Nearest Neighbor (KNN) .....	34
4.4 Random Forest.....	40
4.5 Tabella riassuntiva .....	46
<b>5. Conclusione e lavori futuri</b> .....	<b>47</b>
<b>Bibliografia</b> .....	<b>49</b>

## ABSTRACT

Per la grande rilevanza che sempre di più assume la questione ambientale, considerando la costante crescita della popolazione mondiale e il conseguente aumento della complessità della gestione dei rifiuti, saper distinguere prodotti biodegradabili da quelli che non lo sono diventa di fondamentale importanza. Lo studio si occupa di creare, analizzare e comparare diversi classificatori binari con l'obiettivo finale di saper riconoscere, data un'immagine in input, se un prodotto sia o meno dotato della caratteristica della biodegradabilità. Il Machine Learning (ML), sottoinsieme dell'intelligenza artificiale (AI), si occupa di creare sistemi intelligenti che acquisiscono e migliorano le performance in base ai dati di cui vengono disposti. Un corretto utilizzo di tale intelligenza, intrecciata con una efficiente raccolta di dati immessa poi nel processo, ha restituito delle performance di un certo rilievo per tale scopo.

Il dataset utilizzato è il Non and Biodegradable Material Dataset, ottenuto dal sito Kaggle.

Il task affrontato è stato il frutto di metodi di machine learning quali Support Vector Machines SVM, Random Forest e K- nearest neighbor (KNN), con tecniche di pre-processing quali filtri come il Sobel, l'Entropy e la Discrete Fourier Transform (DFT) e l'analisi dello spazio RGB con lo scopo di effettuare feature selection.

I risultati mostrano il random forest come il modello migliore addestrato lavorando con immagini nel dominio della frequenza applicando sulle stesse la trasformata discreta di Fourier. A seguire il modello support vector machine è quello che si comporta meglio in media, risultando molto vicino al random forest. Discorso diverso invece per il K- nearest neighbor che fornisce i risultati peggiori, anche se con qualche eccezione che però non eguaglia mai gli altri due modelli.

# 1 INTRODUZIONE

Ogni anno la popolazione mondiale aumenta di circa l'1.1%. Ciò comporta molteplici conseguenze, tra le quali un impatto ambientale non indifferente. Diventa di fondamentale importanza saper gestire i rifiuti che produciamo. Per questo motivo si rende necessario marcare la differenza tra due tipi di rifiuto: biodegradabile e non-biodegradabile.

Un prodotto si dice biodegradabile se disperso nell'ambiente, si decompone facilmente in composti meno inquinanti, grazie all'azione di batteri o altri microorganismi. Esempi di prodotti biodegradabili sono: gli scarti di verdure, carne, gusci d'uovo, ecc. Questi rifiuti contengono nutrienti necessari alla crescita delle piante. Infatti, la biodegradazione riveste un ruolo fondamentale nel mantenere l'equilibrio ecologico degli ecosistemi ed in generale del pianeta. In contrapposizione abbiamo i prodotti non-biodegradabili, che non vengono in alcun modo scomposti per essere assorbiti dal terreno rimanendo immutati per molto tempo, tempo durante il quale contribuiscono all'inquinamento dell'area in cui vengono a trovarsi. Esempi classici di prodotti non biodegradabili sono le bottiglie di plastica o di vetro, la prima impiega dai 100 ai 1000 anni per decomporsi mentre, la seconda, ne impiega anche 4000.

Considerando il fatto che in media ogni cittadino (solamente in Italia) produce circa 500 chilogrammi di spazzatura ogni anno, e tra di esse ci sono sia prodotti biodegradabili che non, è rilevante riuscire a distinguere a livello visivo quelli che effettivamente sono biodegradabili da quelli non, per permettere anche una raccolta differenziata più efficiente.

La classificazione delle immagini di prodotti o rifiuti biodegradabili può essere effettuata con diversi approcci, principalmente due:

- Machine learning, il quale è una branca dell'intelligenza artificiale che si occupa di costruire, analizzare e implementare algoritmi che permettono ai computer di apprendere e adattarsi utilizzando modelli statistici per analizzare e trarre inferenze da modelli nei dati a partire da feature individuate da esperti di dominio.

- Deep Learning, il quale è un sottoinsieme del machine learning come è possibile notare in figura 1, è essenzialmente una rete neurale con uno o più livelli che tentano di simulare il comportamento del cervello umano, cercando di imparare da grandi quantità di dati. La differenza principale con il machine learning è che in genere richiede meno pre-processamento dei dati, automatizzando l'estrazione delle feature.

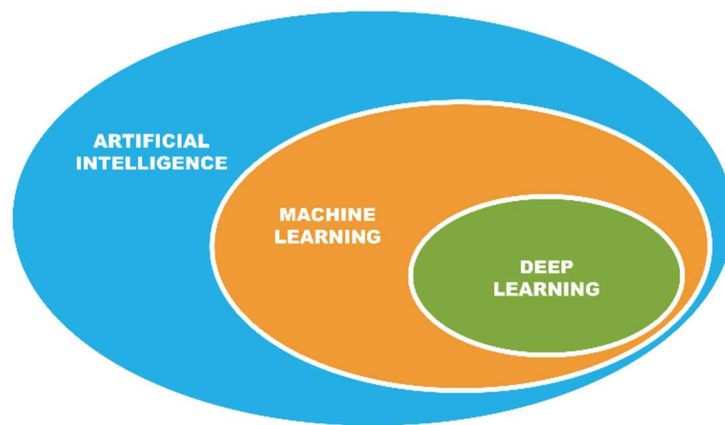


Figura [1]: il machine learning non è altro che una branca del vasto settore dell'intelligenza artificiale, mentre il deep learning è a sua volta una sottocategoria del machine learning.

In letteratura, la classificazione di immagini biodegradabili e non è spesso effettuata utilizzando metodi di deep learning, soprattutto usando una CNN (Convolutional Neural Network), un particolare tipo di rete neurale che prende ispirazione dal funzionamento dei neuroni, particolarmente adatta per classificazioni di immagini richiedendo un pre-processing inferiore rispetto ai metodi di machine learning tradizionali.

Tuttavia le metodologie utilizzate in questa tesi per compiere questo task seguono un approccio orientato al machine learning tradizionale, di seguito l'elenco dei modelli utilizzati in questo studio:

1. Support-Vector-Machine (SVM): presentato nel capitolo 2.1;
2. K- nearest neighbor (KNN): presentato nel capitolo 2.2;
3. Random Forest (RF): presentato nel capitolo 2.3.

Una volta presentati i metodi utilizzati si prosegue con la descrizione dei metodi usati per fare feature extraction, ovvero estrarre le informazioni più rilevanti dalle varie immagini del dataset. In particolare sono stati utilizzati filtri come la Discrete Fourier Transform DFT, l'entropia, e il Sobel. Infine è stata fatta un'analisi dello spazio RGB per cercare di capire il canale per distinguere in maniera più accurata tra prodotti biodegradabili e non.

I risultati di tutte queste analisi sono poi riportati nel capitolo 4, mentre le conclusioni e una descrizione dei prossimi passi è stata presentata nel capitolo 5.

## 2 ALGORITMI DI MACHINE LEARNING

Il machine learning è la scienza che studia gli algoritmi e i modelli statici che permettono ad un computer di eseguire dei task senza essere esplicitamente programmati per quel determinato compito. [4]

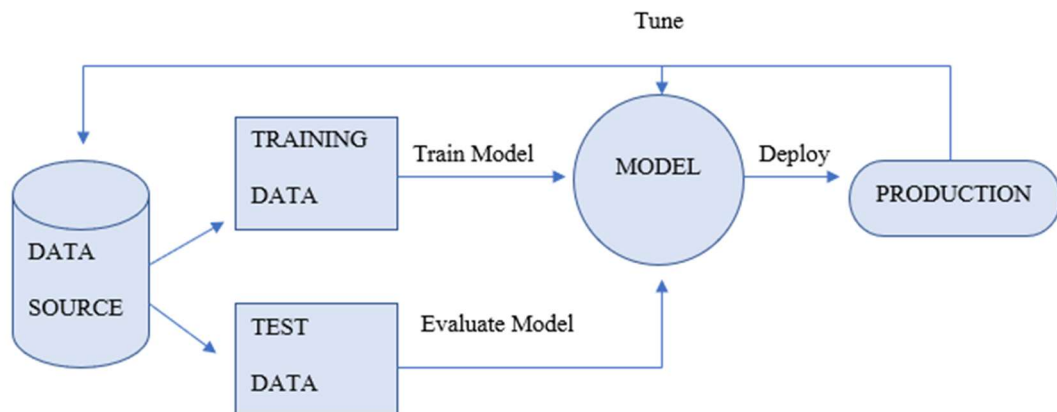


Figura [2]: Uno schema riassuntivo di ciò che avviene nel machine learning

Come è riassunta nella figura 2 il processo inizia dal dataset da cui si vuole apprendere. Esso viene diviso in due set: Training data, ovvero i dati utilizzati da modello per “imparare”, ed il test data, i dati su cui il modello effettua test una volta che ha terminato il processo di apprendimento. Tutto ciò è scandito da una fase di intermezzo, il tuning dei parametri, procedimento attraverso il quale si testano differenti parametri per cercare la combinazione che faccia migliorare l’accuratezza e la robustezza del modello. Questa fase è stata descritta in dettaglio per ogni modello nei seguenti capitoli: 2.3 Support Vector Machine, 2.4 K- nearest neighbor, 2.5 Random Forest, e successivamente nel capitolo 3 sono stati utilizzati i risultati migliori.



## 2.1 SUPPORT VECTOR MACHINE (SVM)

Uno degli algoritmi utilizzati per risolvere questo task è stato il Support Vector Machine (SVM) [1], un algoritmo di apprendimento supervisionato utilizzato sia per problemi di regressione lineare che per problemi di classificazione, come è stato usato in questo caso specifico.

Alcune definizioni utili per capire:

- Support vectors: i punti che sono più vicini all'iperpiano;
- Iperpiano: un sottospazio lineare di dimensione inferiore a 1 rispetto allo spazio in cui sono contenuti i dati. Ad esempio se i nostri dati vivono in uno spazio 3-dimensionale l'iperpiano sarà un piano 2d, se invece abbiamo dei dati in 2d l'iperpiano sarà semplicemente una retta;
- Margine: la distanza tra l'iperpiano e i punti più vicini da entrambi i lati;
- Kernel: una funzione matematica usata per trasformare i dati di input in forma differente.

Come funziona è riassunto nella figura 3: lo scopo è quello di trovare l'iperpiano che divida i dati nel miglior modo possibile. Per "migliore" si intende un iperpiano con i margini maggiori, in modo tale da rendere la separazione più "netta" possibile.

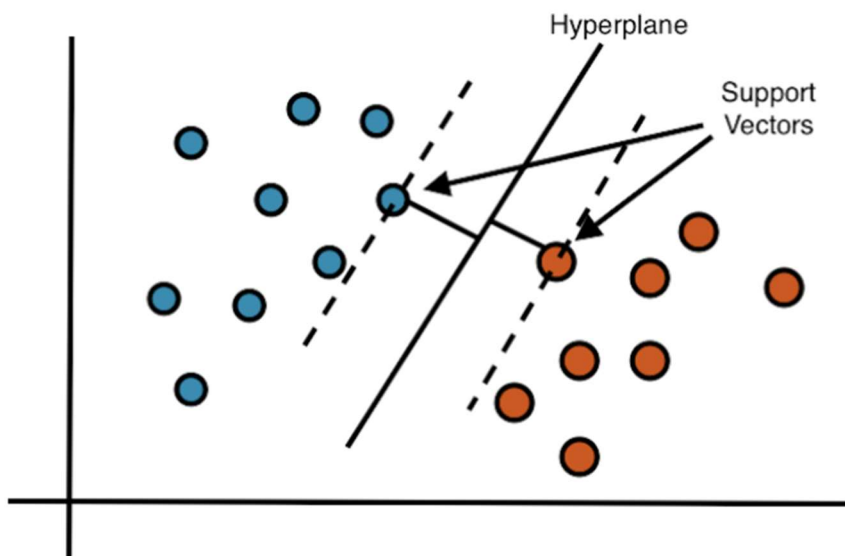


Figura [3]: Uno spazio 2d con una retta (iperpiano) che divide i dati "blu" dai dati "rossi".

I kernel più usati sono linear, polynomial, rbf e sigmoid, ma questo si vedrà in seguito.

I vantaggi del SVM sono sicuramente la versatilità, l'efficienza dal punto di vista della memoria e la sua efficacia nel caso il numero di dimensioni sia maggiore del numero di sample. Tuttavia, tale modello non è esente da svantaggi, quale l'overfitting, il costo computazionale e la difficoltà di trovare una spiegazione probabilistica dei risultati. La scelta dei parametri quindi risulta fondamentale per evitare problemi tipo l'overfitting o l'underfitting, anche se per effettuare una scelta sensata bisogna prima capire cosa rappresenta ognuno di esso, di conseguenza di seguito sono elencati i diversi parametri e la loro spiegazione

1. **C:** Un SVM standard cerca di separare le varie classi di dati "nettamente", non ammettendo nessuna errata classificazione, si ottiene quindi un modello abbastanza legato ai dati, rendendolo estremamente sensibile al rumore provocando overfitting. Per questo motivo, si cerca di rendere questo confine più "morbido", ammettendo qualche errata classificazione come è mostrato nella figura 4. Ed è qui che entra in gioco il parametro  $C$ , che regola la "penalità" per ogni errata classificazione, cioè se  $C$  è grande darà una grande penalità ad ogni errore di classificazione, ottenendo una separazione più "rigida" (e tendendo verso l'overfitting), mentre più  $C$  è piccolo meno penalità da ad ogni errore, di fatto ammettendo più tolleranza, rendendo in questo modo il modello più generale, astratto.

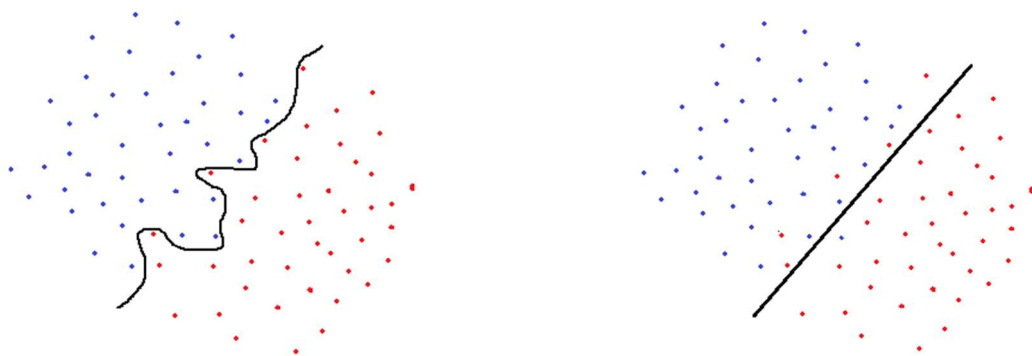


Figura [4]: a sinistra abbiamo un SVM standard che non ammette errori, mentre a destra abbiamo un SVM il cui  $C > 0$  e come possiamo osservare ci sono alcuni punti erratamente classificati.

2. **KERNEL:** In alcuni casi i dati in input non sono così facili da separare, quindi bisogna prima “trasformarli” in una forma per rendere lo scopo più facilmente raggiungibile come illustrato dalla figura 5. Per fare ciò, si usa una funzione kernel, e le più importanti sono 3.

- a. linear -> usato da SVM standard quando è possibile separare linearmente i dati. La sua equazione è la seguente:

$$K(x, x_i) = \text{sum}(x * x_i)$$

- b. Radial Basis Function kernel (rbf), il più utilizzato quando i dati non sono linearmente separabili, esprimibile con la seguente formula:

$$K(x, x_i) = \exp(-\gamma * \text{sum}((x - x_i)^2))$$

- c. Polynomial Kernel, è la forma generalizzata del kernel linear, anche esso utilizzato per dati non linearmente separabili, ed è espresso secondo la seguente formula:

$$K(x, x_i) = 1 + \text{sum}(x * x_i)^d$$

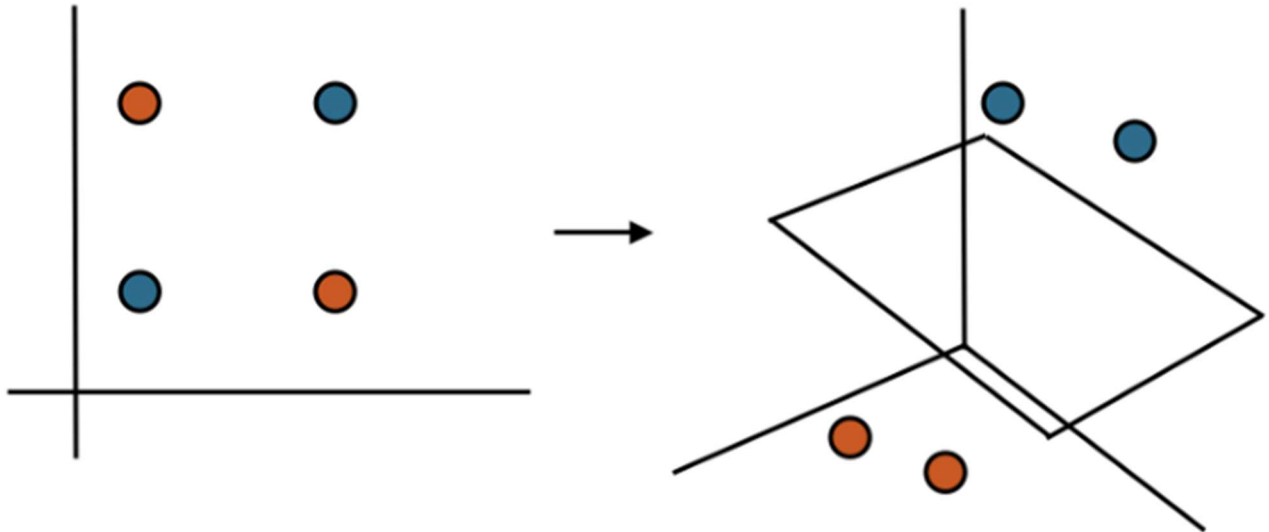


Figura [5]: Dal momento in cui non c'è un modo logico di suddividere i dati nello spazio in cui vivono, si trasforma lo spazio in input da 2d a 3d. A questo punto, come possiamo vedere in figura, i dati diventano separabili molto facilmente.

## OSSERVAZIONE

Per quanto riguarda i kernel non lineari, essi non trasformano i punti in nuovi punti appartenenti ad uno spazio dimensionale più grandi, ma calcolano l'iperpiano in base alla somiglianza di questi punti misurata nello spazio dimensionale maggiore. Questo punto è fondamentale per capire bene il lavoro del parametro gamma

3. **GAMMA:** questo parametro viene utilizzato per controllare i kernel non lineari. Ad esempio con rbf, gamma controlla la grandezza della distanza che devono avere due punti nello spazio dimensionale maggiore per essere considerati simili. Di conseguenza con bassi valori di gamma questa distanza è piccola, quindi molti punti vengono raggruppati insieme (si tende a generalizzare molto, possibile underfitting), invece con alti valori di gamma i punti devono essere molto vicini per essere raggruppati (si tende all'overfitting) come possiamo osservare dalla figura 6.

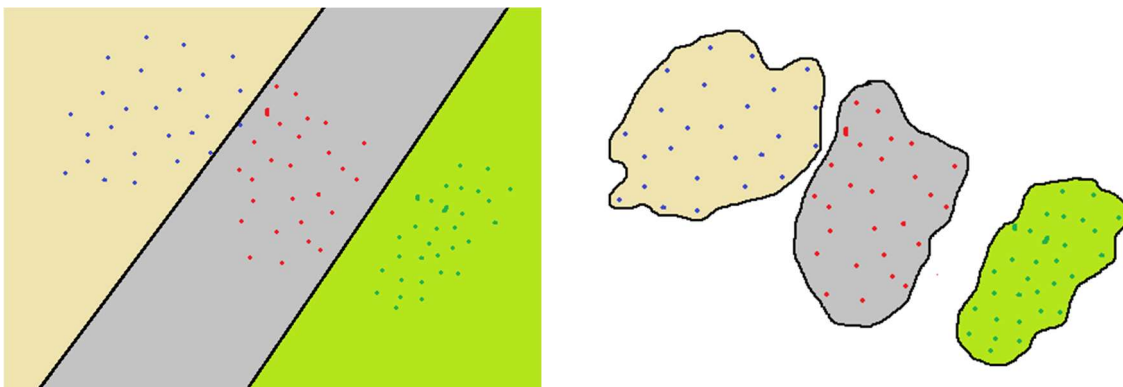


Figura [6]: nell'immagine a sinistra si ha un valore di gamma basso, mentre nella figura a destra si ha un valore di gamma elevato.

I risultati migliori ottenuti aggiustando questi parametri possono essere rinvenuti nel capitolo 4.2.

## 2.2 K-NEAREST NEIGHBOR

Il K- nearest neighbor (KNN) è un altro tipo di algoritmo appartenente alla categoria supervised learning, usato principalmente per scopi di classificazione [2].

L'idea alla base di questo modello è molto semplice: dato in input un dataset  $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$  cui ogni dato è fornita una label ( $y_i$ ) che indichi la classe di appartenenza (supervised learning) il K- nearest neighbor lo memorizza e successivamente per ogni nuovo dato fornitogli in input non etichettato calcola la distanza tra tutti gli altri punti e trova i k più vicini al suddetto punto, e lo classifica in base alla classe maggiormente presente tra essi. Un esempio di classificazione utilizzando questo modello è rinvenibile in figura 7.

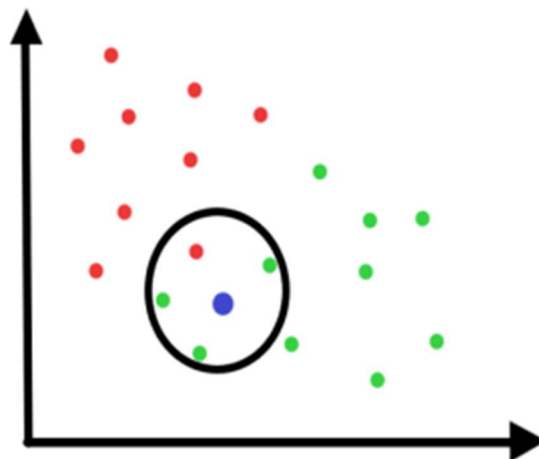


Figura [7]: Un esempio di K- nearest neighbor dove abbiamo 2 tipi di classi rosso e verde e si vuole classificare un nuovo dato, in blu. In questo caso, se si considerano i primi 4 vicini si hanno 3 verdi e 1 rosso, quindi il modello classificherà il nuovo dato blu come appartenente alla categoria dei verdi.

I parametri più importanti per questo modello sono i seguenti:

1. **Distanza:** il tipo di distanza che viene calcolato quando si deve verificare la distanza del punto da classificare da tutti gli altri, le più usate sono:
  - a. Distanza Euclidea: calcolata come la radice quadrata della somma delle differenze tra un nuovo punto ( $x_i$ ) e i punti nel dataset ( $y_i$ )

$$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

- b. Distanza di Manhattan: altra metrica molto utilizzata, viene calcolata facendo il valore assoluto della differenza tra i punti.

$$\sum_{i=1}^k |x_i - y_i|$$

- c. Distanza di Minkowski: questa metrica è la generalizzazione delle 2 precedenti, infatti con il parametro  $p=1$  corrisponde a quella di Manhattan mentre con  $p=2$  a quella Euclidea, con  $p>2$  si creano altre metriche di distanza.

$$\left( \sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}$$

2. **K**: numero di vicini da considerare per effettuare la predizione.

Il valore di K è importante nel definire i “confini” fra le diverse categorie di dati. Un valore grande di K può aiutare nella riduzione del “rumore”, mentre un valore troppo piccolo di K renderebbe il modello troppo sensibile ad esso.

Ad esempio, se si sceglie  $K = 1$ , il modello considererà soltanto il dato più vicino, che nel caso del training set sarà la classe del dato stesso, ritrovandoci quindi in una situazione di overfitting.

Riassumendo

I vantaggi del K- nearest neighbor principali sono:

- Di facile implementazione;
- Non ha un tempo di training, siccome i dati stessi sono il riferimento per la previsione futura. Di conseguenza, in termini di tempo, è molto performante.

Tra gli svantaggi si ha:

- Funziona in modo errato con dataset molto ampi, siccome bisogna calcolare la distanza tra ogni coppia di punti;
- Non funziona bene con dati ad alta dimensionalità;
- Sensibilità al rumore nei dati.

Come per il support vector machine, anche per il K- nearest neighbor sono state effettuati molteplici test per individuare i parametri migliori, reperibili nel capitolo 4.2

## 2.3 RANDOM FOREST

Prima di definire cosa sia il random forest, bisogna partire dal concetto di decision tree. Quest' ultimo è un modello di machine learning che può essere rappresentato come un albero. [7] Esso si basa sul fatto che ogni dato del dataset può descritto come un insieme di attributi, ognuno di essi delimitato da un range discreto di valori assumibili. Di conseguenza, per classificare un oggetto si parte dalla radice dell'albero e per ogni nodo, in base al valore di quella proprietà, viene presa una determinata direzione. Questo processo continua fino a che non viene raggiunto un nodo foglia, come nell'esempio mostrato nella figura 8.

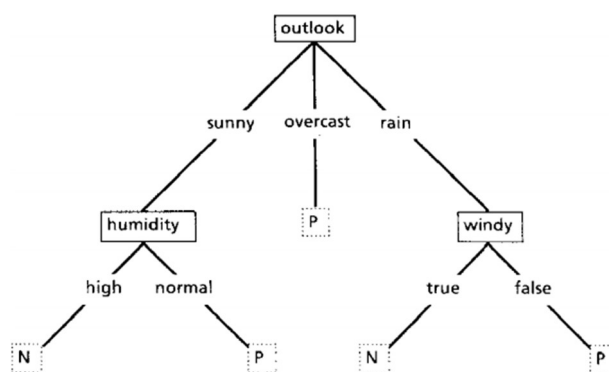


Figura [8]: Un esempio di albero di decisione che predice se un determinato dato appartiene alla classe P o N (positive o negative)

Il random forest non è altro che un insieme di decision trees individuali, che sono utilizzati in ensemble, ovvero combinati tra loro per ottenere una migliore predizione rispetto a quella restituita dai singoli.

Questo modello, come si vedrà poi nei test, è quello che restituisce il risultato migliore, seppure di poco distante da quello migliore restituito dal SVM, e ciò concorda con quanto trovato in letteratura dato che il random forest è considerato un modello molto performante. La chiave del suo successo è la bassa correlazione tra i tanti decision trees di

cui è formato, che cooperano insieme per formare un modello individuale.

Concettualmente è semplice: mentre alcuni alberi possono fare una predizione sbagliata, molti altri possono fare una scelta giusta che può portare il modello sulla retta strada.

Per mantenere una bassa correlazione tra i differenti decision trees, viene sfruttata una debolezza di quest' ultimi essendo molto sensibili al rumore. Nel concreto, piccoli cambiamenti influiscono molto sulla loro struttura, e ad ognuno di essi viene fatto campionare un subset casuale dal dataset, formando così alberi diversi fra loro. Questo procedimento è noto come bagging.

I parametri per questo modello sono i seguenti:

- a. **n\_estimators**: questo parametro indica il numero di decision trees che si vuole utilizzare. Un numero elevato porterebbe ad un modello con meno varianza e più robustezza, ma impatterebbe in termini di tempo. Se abbiamo un numero alto di feature e numero basso di n\_estimators molte di esse, magari importanti, potrebbero essere ignorate;
- b. **criterion**: questo parametro indica la regola secondo la quale un decision tree calcola quale feature e quale valore di essa suddivide meglio le osservazioni fatte fino a quel momento. Le regole più importanti sono 2:

$$\text{I.} \quad \text{Gini} = 1 - \sum_{j=1}^c p_j^2 = 1$$

$$\text{II.} \quad \text{Entropy} = - \sum_{j=1}^c p_j \log p_j = 1$$

- c. **Max-depth**: indica la massima profondità di un decision trees, la quale aumentandola incrementa il numero di feature prese in considerazione. Più profondo è l'albero, più split vengono effettuati, e di conseguenza si avranno più informazioni. Nel decision tree questo aumento causa overfitting, ma per il discorso fatto in precedenza è più difficile fare overfitting del random forest, anche se non si esclude che con valori molto grandi ciò possa accadere.
- d. **max\_features**: come anticipa il nome, è il numero di feature casuali da considerare ad ogni split. Sebbene sia il più difficile da ottimizzare, esso è uno dei parametri più importanti,. Un valore piccolo di questo parametro può ridurre la varianza, e allo stesso tempo può incrementare il bias portando ad underfitting.



Al contrario, un valore troppo alto aumenta la varianza e favorisce l'overfitting. I possibili valori sono:

- a. **Sqrt**: viene presa la radice quadrata del numero totale di feature;
- b. **None**: vengono considerate tutte le feature;
- c. **Log2**: viene preso il logaritmo in base 2 del numero totale di feature.

Come riportato in [8] si afferma che il random forest fornisce ottimi risultati anche con i settings di default, e che la "tunability" di questo algoritmo è molto minore rispetto agli altri, come ad esempio SVM. Ciò che si afferma in letteratura rispecchia poi i risultati ottenuti nel capitolo 4, tuttavia durante l'analisi si è cercato di ottimizzare comunque i parametri.

### 3. SOLUZIONI PROPOSTE

In questo capitolo sono descritte e analizzate le varie tecniche per estrarre le informazioni più rilevanti dalle varie immagini del dataset al fine di migliorare le prestazioni e le accuratezze dei vari modelli. Ciò è svolto tramite filtri di Sobel descritto nel capitolo 3.2, la trasformata discreta di Fourier descritta nel capitolo 3.3 e infine l'entropia descritta al capitolo 3.4. Si passa successivamente all'analisi dello spazio dei colori, e infine è proposta la Principal Component Analysis per visualizzare i dati.

#### 3.1 FEATURE SELECTION

Uno dei maggiori problemi che si affronta quando si tratta di creare modelli di classificazione sono le performance, soprattutto quando il dataset trattato è vasto come in questo caso.

Per questo motivo è utile distinguere i dati utili da quelli meno utili in modo tale da ridurre il carico e velocizzare il processo di apprendimento del modello.

Feature selection è parte del processo di riduzione dimensionale, dove un insieme iniziale di dati viene ridotto in un gruppo più maneggevole, tecniche di questo tipo sono analizzate e descritte nei seguenti capitoli.

#### 3.2 EDGE DETECTION

L'edge detection è un'operazione che ci permette di processare le immagini con lo scopo di ridurre la quantità di pixels ma allo stesso mantenere la struttura delle immagini invariata. Tale operazione consiste nel trovare le regioni dell'immagine dove si ha un brusco cambiamento di colore. Queste regioni corrispondono con i contorni degli oggetti rappresentati in figura.

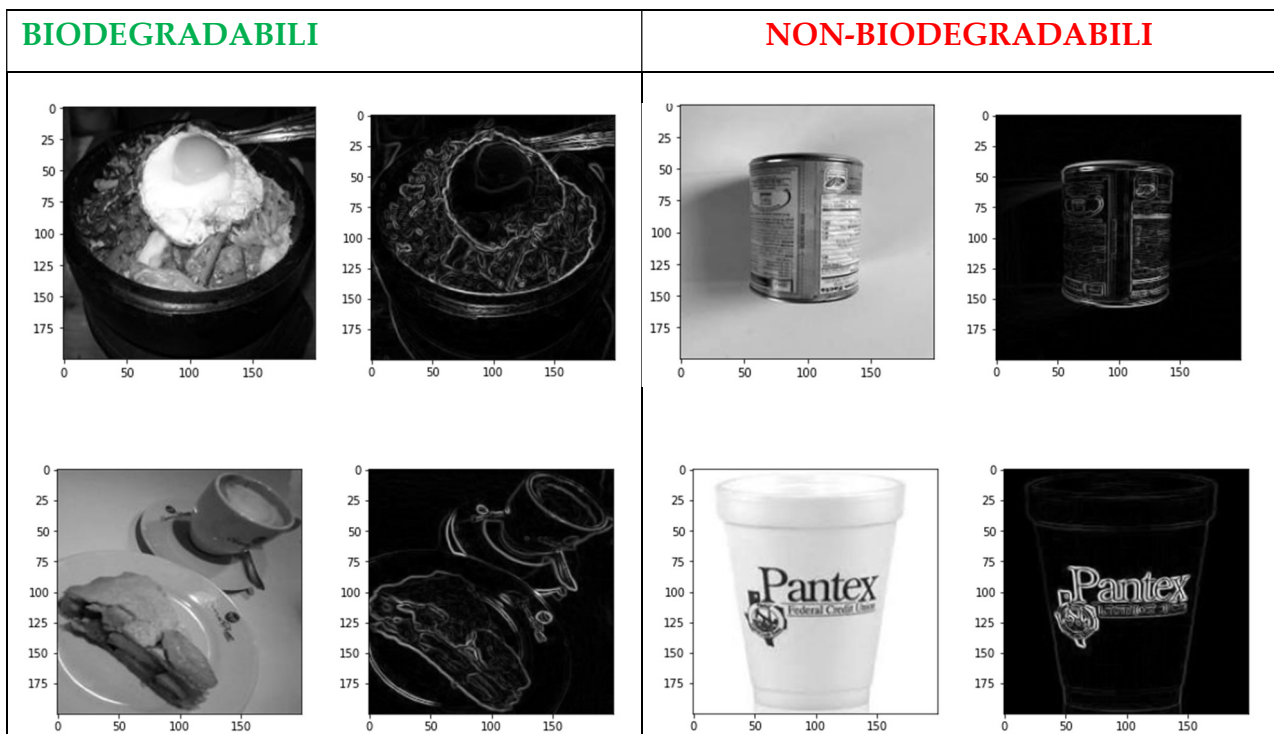
Ci sono svariati modi di implementare l'edge detection, uno dei quali è l'operatore Sobel, utilizzato in questo studio. Si tratta di un algoritmo che calcola un valore approssimato del gradiente in ciascun punto dell'immagine, trovando la direzione lungo la quale si ha

l'incremento maggiore dal chiaro allo scuro, che corrisponde ad un'alta probabilità che quella parte di immagine rappresenti un bordo, fornendo anche una stima del suo orientamento. In termini matematici, l'operatore applica due matrici di convoluzione 3x3 all'immagine una in direzione orizzontale ed una in direzione verticale come mostrato in figura 9.

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{e} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

Figura [9]: A l'immagine originale, Gx Gy due immagini i cui punti rappresentano rispettivamente i valori approssimati delle derivate in orizzontale ed in verticale

Di seguito possiamo notare alcune immagini appartenenti ad entrambe le categorie a cui è stato applicato l'operatore sobel. Come ben si nota il filtro lavora egregiamente individuando i contorni in modo efficiente. I risultati ottenuti addestrando i diversi modelli con questo particolare filtro sono ottenibili nei capitoli 4.2, 4.3, 4.4 per i diversi tipi di modelli di machine learning.



## OSSERVAZIONE

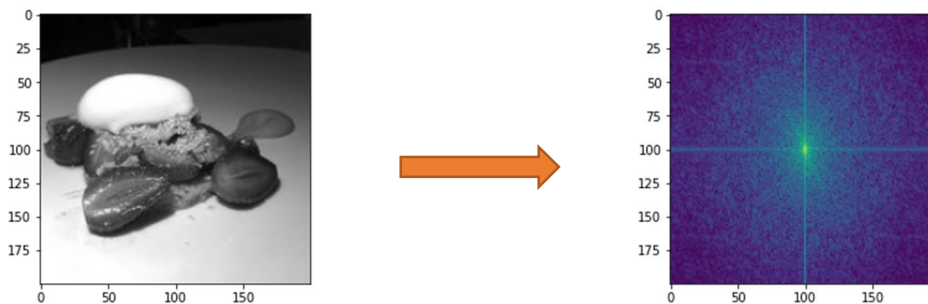
Non si notano particolari caratteristiche ad occhio nudo che facciano immediatamente distinguere le immagini biodegradabili da quelle non biodegradabili.

### 3.3 DISCRETE FOURIER TRANSFORM

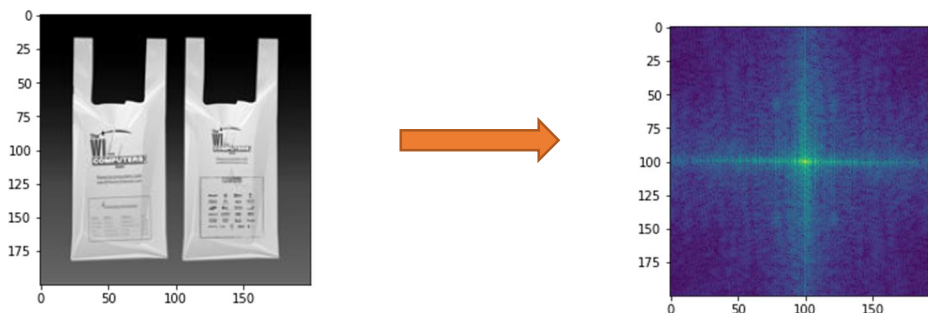
La trasformata di Fourier è una tecnica basata sulla scomposizione dei segnali in somme sinusoidi. La DFT è una particolare trasformata di Fourier e si definisce come la successione di  $N$  numeri complessi  $x_0, x_1, \dots, x_{N-1}$  trasformata in un'altra successione di numeri complessi  $X_0, X_1, \dots, X_{N-1}$  secondo la seguente formula

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn}, \quad k = 0, \dots, N-1,$$

Di seguito possiamo osservare un'applicazione della DFT su un'immagine:



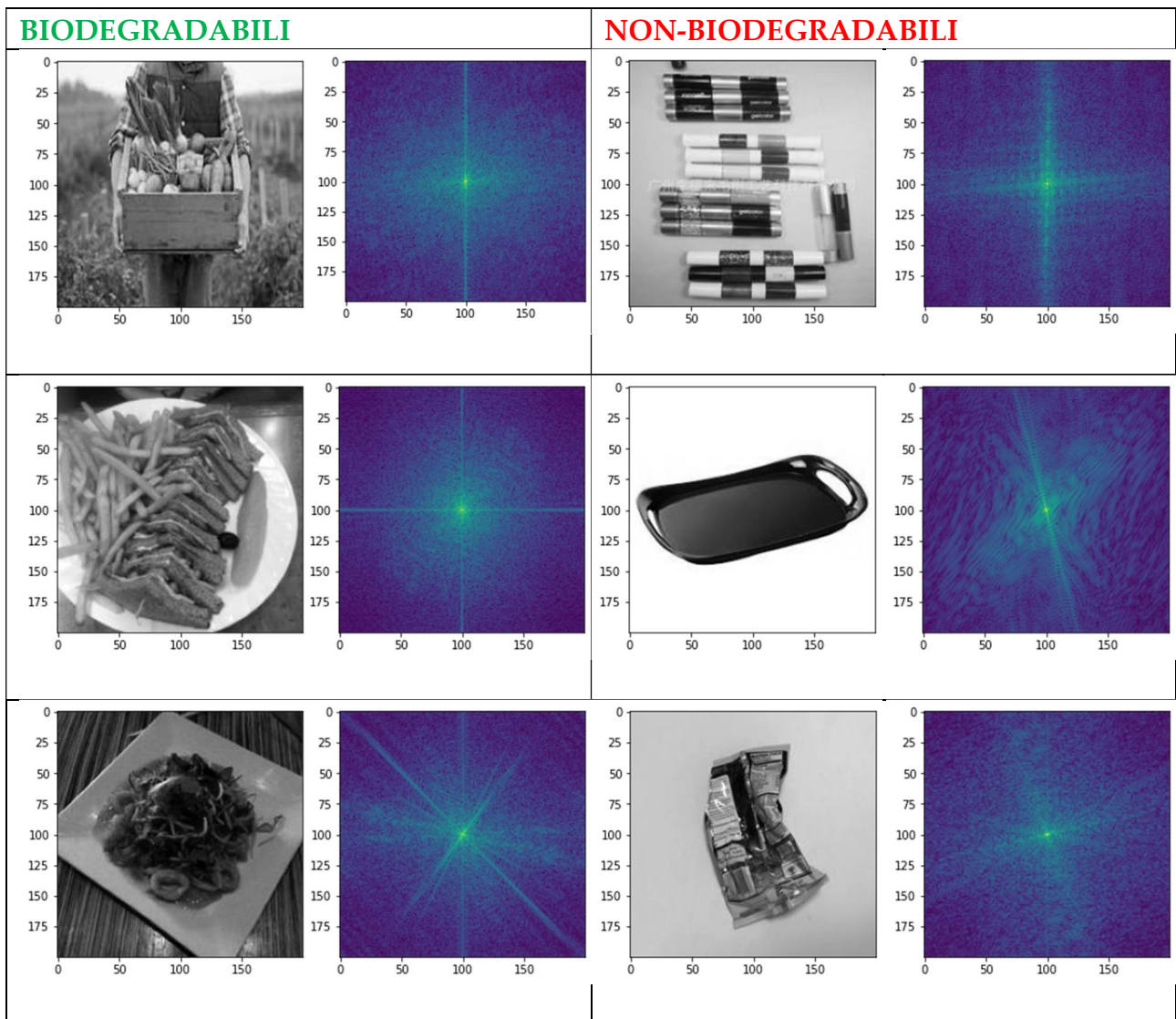
Qui possiamo notare un'immagine biodegradabile in scala di grigi a sinistra e la sua DFT applicata a destra. Al centro della DFT abbiamo le frequenze basse mentre avvicinandoci sempre di più ai margini abbiamo le frequenze più alte.



Qui invece, si può vedere un esempio di un'immagine non-biodegradabile a cui è applicata la DFT.

## OSSERVAZIONE:

Nelle immagini biodegradabile la dft sembra essere più “definita” mentre nelle immagini non-biodegradabili sembra invece più “rumorosa” (DA SCRIVERE MEGLIO), ecco in seguito alcuni esempi di sample di immagini random che hanno innescato questa osservazione:



La DFT quindi è utilizzata per trasformare un'immagine dal dominio dei pixel al dominio della frequenza, una volta fatto ciò, vengono utilizzate queste immagini per trainare il modello. Nel capitolo 4 sono stati effettuati test per verificarne l'utilità di questo approccio che dà risultati veramente sorprendenti, confermando l'intuizione che è stata fatta.

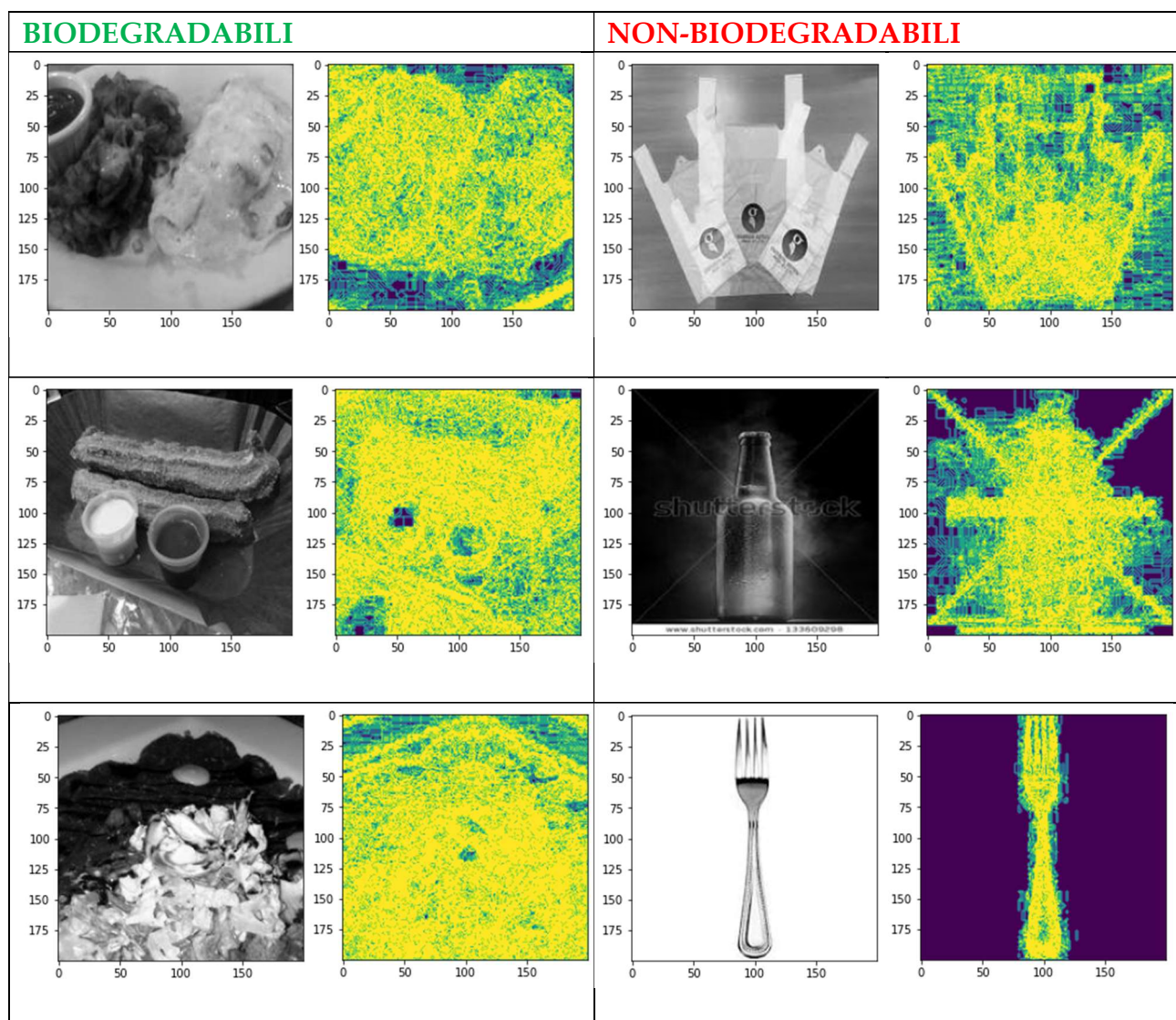


### 3.4 ENTROPIA

Un filtro piuttosto interessante è stato l'entropia. Essa viene interpretata come una misura del disordine presente in un sistema fisico qualsiasi. Per quanto riguarda le immagini, l'entropia indica la complessità contenuta in una determinata area dell'immagine, tipicamente definita da elementi strutturali dell'immagine.

Questo filtro viene usato su immagini in scala di grigi ed è in grado di rivelare anche sottili variazioni nella distribuzione del livello di grigio locale.

Di seguito sample di immagini ottenute applicando l'entropia:

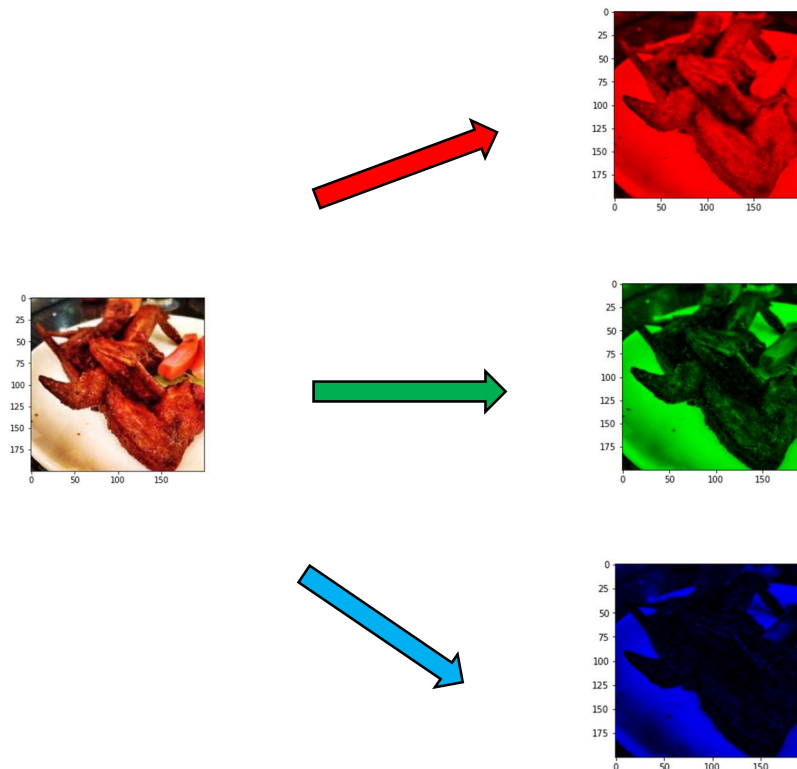


I risultati ottenuti dai diversi modelli dando queste immagini come allenamento sono rinvenibili al capitolo 4.

### 3.5 ANALISI SPAZIO RGB

Ogni immagine digitale è formata da 3 colori, il rosso il verde e il blu. Per potere estrarre le feature più importanti è utile analizzare in che modo influiscano i diversi colori nei modelli. A questo scopo le immagini sono state suddivise in 3 matrici, ognuna delle quali rappresenta un colore.

Un esempio è la seguente immagine presa dal dataset:



Questa è una immagine di un prodotto biodegradabile scomposta nei suoi canali RGB. I risultati ottenuti utilizzando questi tre canali per ogni immagine invece della controparte originale sono descritti nel capitolo 4, ed è stato interessante notare come uno dei canali sia quello che dia più informazioni rispetto agli altri.

### 3.6 Principal Component Analysis (PCA)

Inizialmente può sembrare controintuitivo, ma, nel machine learning avere immagini troppo dettagliate può diminuire l'accuratezza del modello invece che migliorarla. Ciò è dovuto a un problema noto in letteratura come "curse of dimensionality".

Uno degli strumenti più popolari per cercare di ridurre la dimensionalità dati è la Principal Component Analysis, ecco come funziona:

Dati  $n$  punti memorizzati in una matrice  $\mathbf{X} \in \mathbb{R}^{d \times n}$  l'obiettivo è quello di rimpiazzare la matrice data con  $\tilde{\mathbf{X}} \in \mathbb{R}^{k \times n}$  una approssimazione di dimensione inferiore con  $k$  nettamente inferiore a  $n$ .

L'idea è trovare  $k \leq d$  direzioni ortogonali con la varianza maggiore che generino uno sottospazio dei dati  $k$ -dimensionale e successivamente proiettare tutti punti della matrice originale su queste direzioni, cercando di perdere meno informazioni possibili.

Per soddisfare quest'ultimo si vanno a cercare quei vettori  $\mathbf{w}$ , ortogonali tra loro, tali che minimizzano l'errore di proiezioni e contemporaneamente massimizzano la varianza dei punti proiettati. L'idea matematica è possibile osservarla in figura 10.

$$\begin{aligned} & [x_1, x_2, \dots, x_d] \quad x \in \mathbb{R}^n \\ & \downarrow \mathbf{xW}, \mathbf{W} \in \mathbb{R}^{n \times k} \\ & \mathbf{z} = [z_1, z_2, \dots, z_k] \quad \mathbf{z} \in \mathbb{R}^k \end{aligned}$$

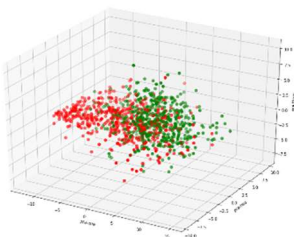
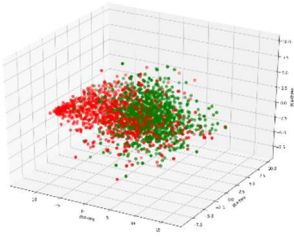
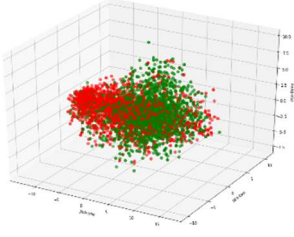
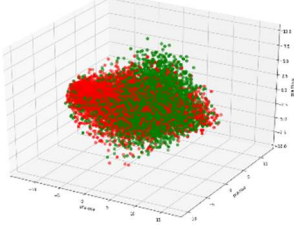
Figura 10: passiamo da un vettore di  $d$  punti a un vettore di  $k$  punti che riesce a perdere il meno informazioni possibili.

Uno dei problemi di questo dataset è che, trattandosi di immagini, i dati vivono in uno spazio dimensionale elevato, per cui diviene difficile visualizzarli. Per questo motivo si utilizzerà la PCA per ridurre la dimensione a 3 in modo da rendere i dati visualizzabili, i risultati ottenuti sono disponibili nella pagina successiva



Legenda:

● Biodegradabile ● Non-Biodegradabile

Numero sample	Plot
500	
1000	
2000	
6000	

## OSSERVAZIONI

Come possiamo notare, i dati tendono a distribuirsi lungo due direzioni diverse.

I punti verdi che rappresentano le immagini biodegradabili tendono a distendersi in modo ortogonale rispetto a come si distendono i punti rossi

## 4. RISULTATI

### 4.1 Dataset

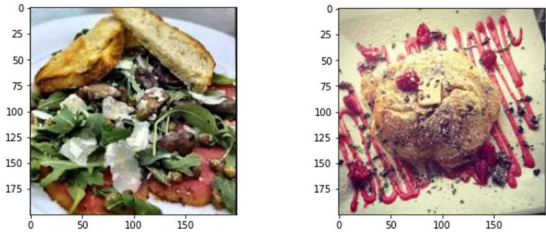
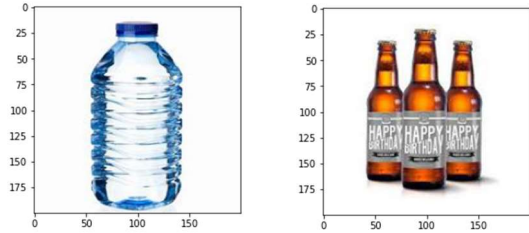
Il dataset raccoglie esattamente 239 690 immagini, tra cui 119 845 rappresentanti biodegradabili, e 119 845 rappresentati prodotti non biodegradabili con risoluzione 200x200 ed è reperibile al seguente link [Non and Biodegradable Material Dataset | Kaggle](#).

A prima vista potrebbe risultare un dataset molto equilibrato, tuttavia analizzandolo a fondo si scopre un'ulteriore suddivisione:

	BIODEGRADABILI	NON-BIODEGRADABILI
Immagini originali	119 845	25587
Immagini che hanno subito un capovolgimento orizzontale	0	23557
Immagini che hanno subito un capovolgimento verticale	0	23567
Immagini ruotate in senso orario	0	23567
Immagini ruotate in senso antiorario	0	23567

Questo procedimento è noto in letteratura come data augmentation, una tecnica usata principalmente per il deep learning con lo scopo di migliorare il training della rete neurale. In questo caso, utilizzando solo modelli di machine learning, applicando o meno questa tecnica non ha apportato nessun tipo di beneficio aggiuntivo, come è stato fatto notare anche nei capitoli successivi.

Di seguito alcune immagini prese casualmente dal dataset in questione:

BIODEGRADABILI	NON-BIODEGRADABILI
	

In questo capitolo analizzeremo i molteplici test che sono stati effettuati per i diversi modelli, attraverso 5 metriche riassunte in questa tabella:

Legenda:

TP = True positive    FP = False positive

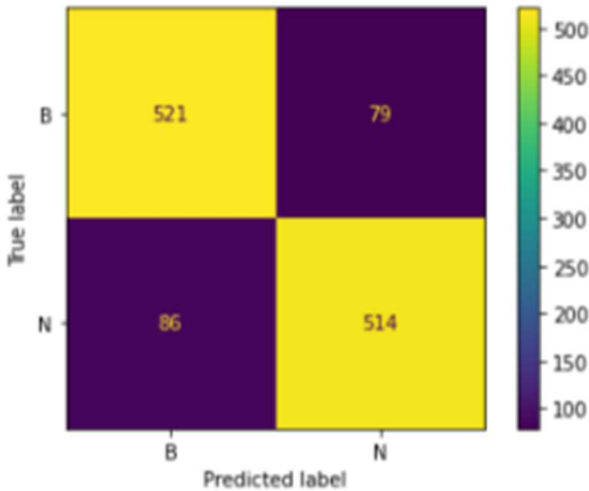
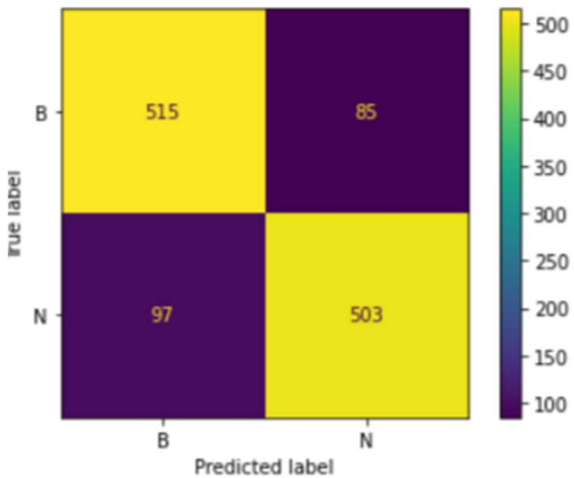
TN = True negative    FN = False negative

Tabella: Le metriche principali utilizzate per valutare le performance dei diversi modelli

Metrica	Rappresentazione matematica	Spiegazione
ACCURACY	$\frac{TP + TN}{TP + FP + FN + TN}$	Uno dei metodi più popolari e lineari, l'accuracy misura quanto spesso il modello è corretto.
SPECIFICITY	$\frac{TN}{FP + TN}$	Misura quanto il modello è prestante nel predire le situazioni negative (nel nostro caso per negativo intendiamo non-biodegradabile)
SENSITIVITY (DETTA ANCHE RECALL)	$\frac{TP}{TP + FN}$	Indica la percentuale di positivi (in questo caso Biodegradabili) correttamente classificati
PRECISION	$\frac{TP}{FP + TP}$	Indica la percentuale dei positivi predetti, realmente positivi
F-SCORE	$\frac{2 * Precision * Recall}{Precision + Recall}$	Le due metriche (precision, sensitivity) non ci danno sufficienti informazioni prese singolarmente, ma combinati possono darci un risultato accurato. A tale scopo ci si serve della metrica F-score che rappresenta la media armonica delle 2 metriche precedenti

## 4.2 SVM

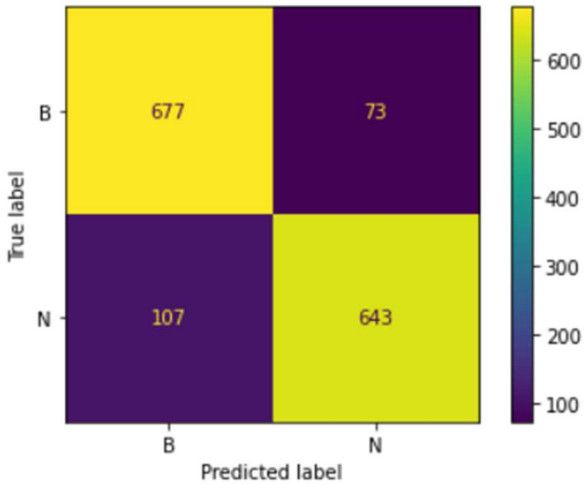
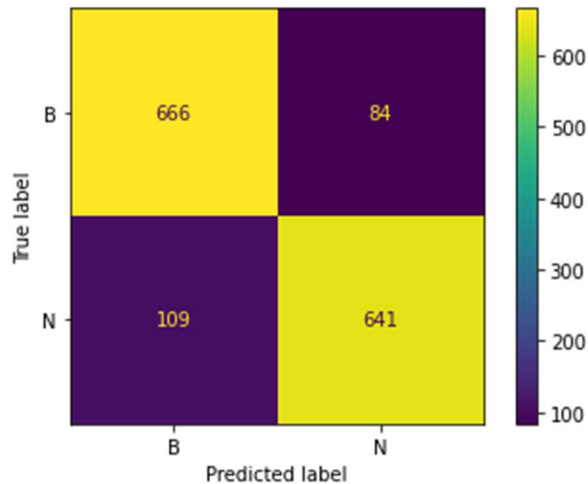
### TEST SVM 01

6000 IMMAGINI SENZA AUGMENTATION	6000 IMMAGINI CON AUGMENTATION SULLA PARTE NON BIO DEGRADABILE
	
<ul style="list-style-type: none"> <li>• Accuracy = 86.25%</li> <li>• Precision = 86.67%</li> <li>• Sensitivity = 85.66%</li> <li>• Specificity = 86.83%</li> <li>• F1-Score = 86.16%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 84.83%</li> <li>• Precision = 85.54%</li> <li>• Sensitivity = 83.83%</li> <li>• Specificity = 85.83%</li> <li>• F1-Score = 84.67 %</li> </ul>
Parametri C = 0.01 Kernel = 'linear'	Parametri C = 0.01 Kernel = 'linear'

Come possiamo osservare i test sono abbastanza simili tra loro, una possibile spiegazione a tutto ciò è perché nonostante ci siano immagini ruotate ciò non influenza granché i diversi modelli, quindi è stato ritenuto inutile riportare le molteplici differenze tra essi per ogni test, di conseguenza nei prossimi test si utilizzeranno dataset formati solo da immagini originali.

## TEST SVM 02

In questo esperimento ci si è concentrati sulla differenza tra i kernel poly e rbf:

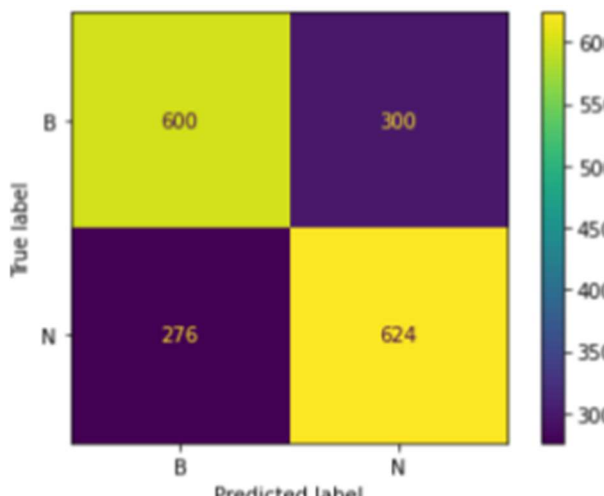
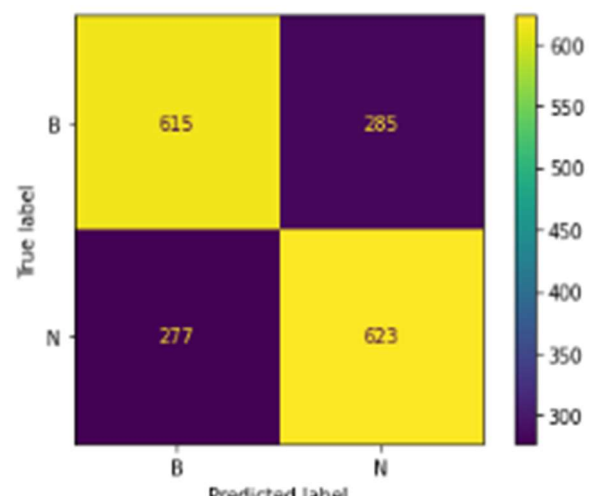
KERNEL RBF	KERNEL POLY
	
<ul style="list-style-type: none"> <li>• Accuracy: 88.0%</li> <li>• Precision: 89.80%</li> <li>• Sensitivity: 85.73%</li> <li>• Specificity: 90.26%</li> <li>• F1-Score: 87.72%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy: 87.13%</li> <li>• Precision: 88.41%</li> <li>• Sensitivity: 85.46%</li> <li>• Specificity: 88.80%</li> <li>• F-Score: 86.91%</li> </ul>
Parametri - C = 1 - Gamma = 0.01	Parametri - C = 0.1 - Gamma = 0.01

Questo test mostra il confronto tra i due kernel non lineari, rbf e poly. Come possiamo osservare il modello dove il parametro kernel è settato a rbf è leggermente migliore di quello poly, e con un C superiore di un ordine di grandezza, ammettendo così più errori di classificazioni, ma più generalizzazione. Gamma invece è risultato essere uguale per entrambi.

## TEST SVM ANALISI RGB

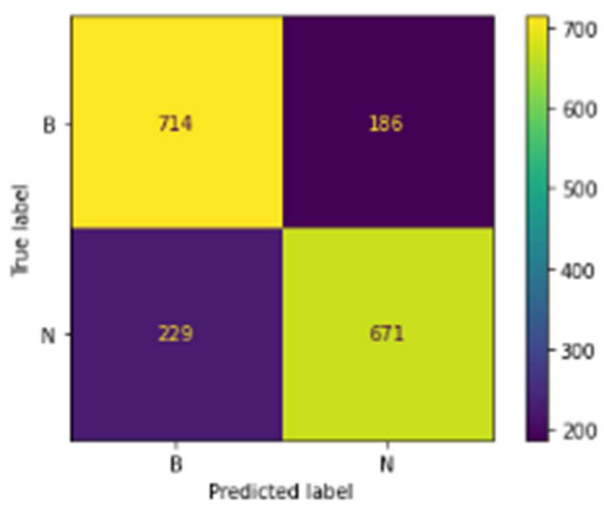
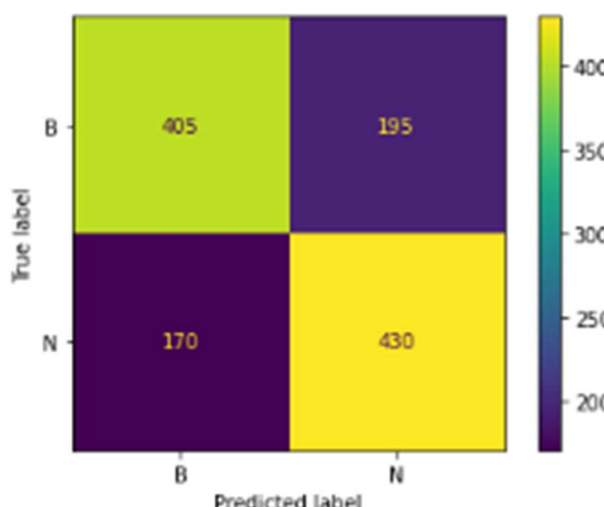
Per tutte e 3 i test sono state utilizzate immagini ridotte a 32x32 e 12000 immagini, 6000 biodegradabili e 6000 non-biodegradabili. Il motivo di tale aumento è che essendo trattato un solo canale le prestazioni aumentano e quindi i tempi diventano minori per il training.

### TEST CANALE ROSSO

KERNEL LINEAR	KERNEL POLY
	
<ul style="list-style-type: none"> <li>• Accuracy = 68.00%</li> <li>• Precision = 67.53 %</li> <li>• Sensitivity = 69.33%</li> <li>• Specificity = 66.66%</li> <li>• F1-Score = 68.42%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy: 68.77%</li> <li>• Precision: 68.61%</li> <li>• Sensitivity: 69.22%</li> <li>• Specificity: 68.33%</li> <li>• F1-Score = 68.91%</li> </ul>
I parametri sono: - C = 0.01	I parametri sono: - C = 0.01 - Gamma = 0.1

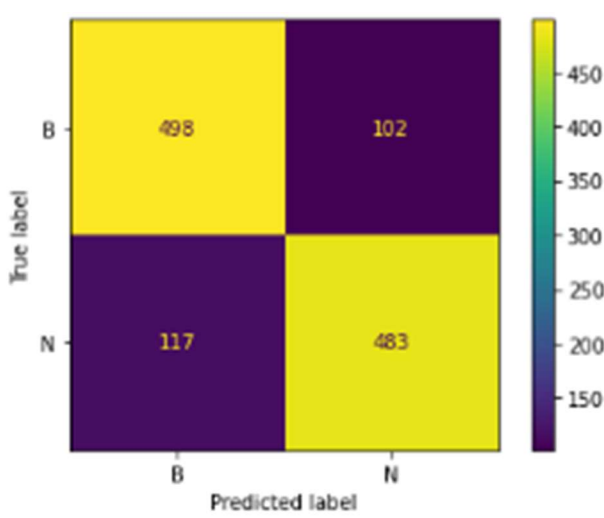
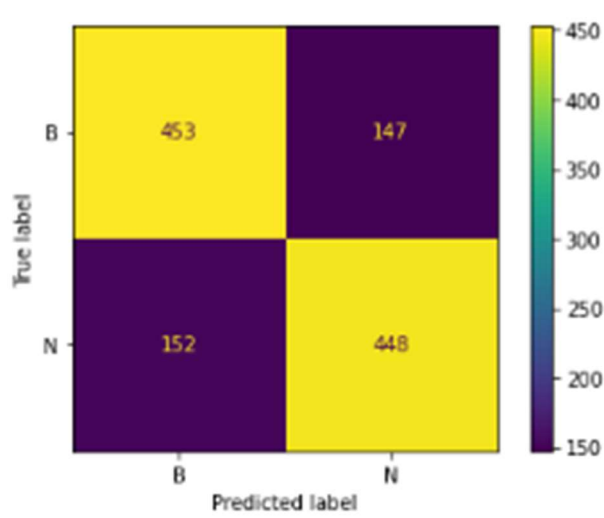
Come evince dai risultati la differenza tra i due kernel è minima seppur leggermente migliori sono i risultati del kernel poly. Ma ciò che è più interessante è che usando solo il canale rosso l'accuratezza diminuisce di molto, e ciò indica che il rosso non ci dà poi così tante informazioni.

## TEST CANALE VERDE

KERNEL LINEAR	KERNEL POLY																		
 <p>Confusion matrix for Kernel Linear:</p> <table><tr><th>True \ Predicted</th><th>B</th><th>N</th></tr><tr><th>B</th><td>714</td><td>186</td></tr><tr><th>N</th><td>229</td><td>671</td></tr></table>	True \ Predicted	B	N	B	714	186	N	229	671	 <p>Confusion matrix for Kernel Poly:</p> <table><tr><th>True \ Predicted</th><th>B</th><th>N</th></tr><tr><th>B</th><td>405</td><td>195</td></tr><tr><th>N</th><td>170</td><td>430</td></tr></table>	True \ Predicted	B	N	B	405	195	N	170	430
True \ Predicted	B	N																	
B	714	186																	
N	229	671																	
True \ Predicted	B	N																	
B	405	195																	
N	170	430																	
<ul style="list-style-type: none"><li>Accuracy: 76.94%</li><li>Precision: 78.29%</li><li>Sensitivity: 74.55%</li><li>Specificity: 79.33 %</li><li>F1-Score: 76.38%</li></ul>	<ul style="list-style-type: none"><li>Accuracy: 69.58%</li><li>Precision: 68.80%</li><li>Sensitivity: 71.66%</li><li>Specificity: 67.50%</li><li>F1-Score: 70.20%</li></ul>																		
I parametri sono: C = 0.1	I paremetri sono: C = 1 Gamma = 0.1																		

Qui è interessante notare che la differenza tra i due kernel si fa più marcata, risultando vincitore il kernel lineare. Tuttavia entrambi i modelli sono migliori di quelli trainati sul canale rosso, se ne deduce quindi che il canale verde ci fornisce più informazioni di quello rosso utilizzando il support vector machine.

## TEST CANALE BLU

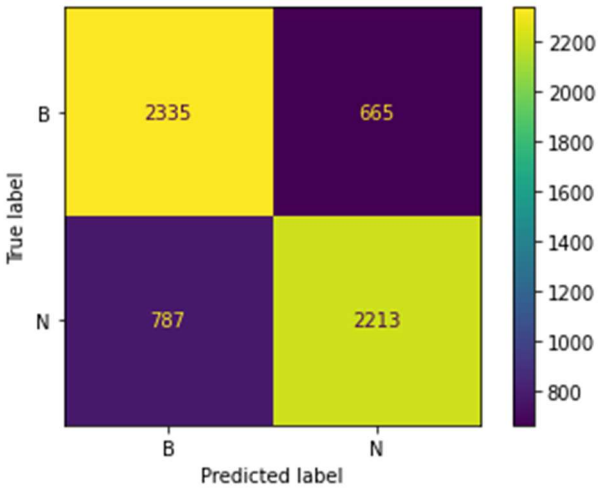
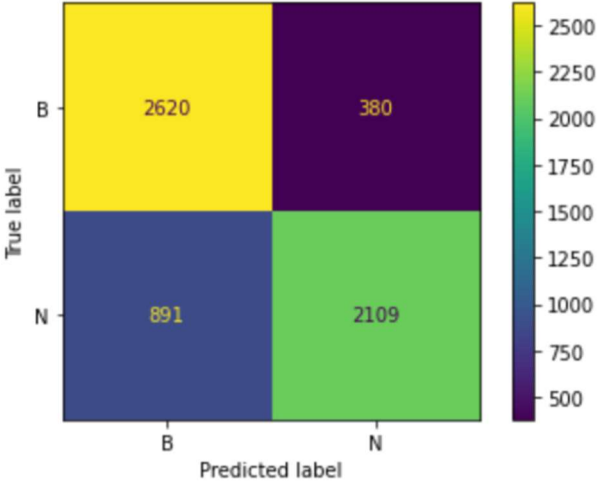
KERNEL LINEAR	KERNEL POLY																		
 <p>Confusion matrix for Kernel Linear model:</p> <table><tr><th>True label \ Predicted label</th><th>B</th><th>N</th></tr><tr><th>B</th><td>498</td><td>102</td></tr><tr><th>N</th><td>117</td><td>483</td></tr></table>	True label \ Predicted label	B	N	B	498	102	N	117	483	 <p>Confusion matrix for Kernel Poly model:</p> <table><tr><th>True label \ Predicted label</th><th>B</th><th>N</th></tr><tr><th>B</th><td>453</td><td>147</td></tr><tr><th>N</th><td>152</td><td>448</td></tr></table>	True label \ Predicted label	B	N	B	453	147	N	152	448
True label \ Predicted label	B	N																	
B	498	102																	
N	117	483																	
True label \ Predicted label	B	N																	
B	453	147																	
N	152	448																	
<ul style="list-style-type: none"><li>• Accuracy: 81.75%</li><li>• Precision: 82.56%</li><li>• Sensitivity: 80.50%</li><li>• Specificity: 83.00%</li><li>• F1-score: 81.51%</li></ul>	<ul style="list-style-type: none"><li>• Accuracy: 75.08%</li><li>• Precision: 75.29%</li><li>• Sensitivity: 74.66%</li><li>• Specificity: 75.50%</li><li>• F-Score: 74.97%</li></ul>																		
I parametri sono: C = 0.1	I paremetri sono: C = 1 Gamma = 0.1																		

Anche in questo test il kernel lineare batte il kernel poly, ma la curiosità è che il blu riesce ad avvicinarsi parecchio ai test ottenuti utilizzando tutte e 3 i canali, di conseguenza è il canale che ci dà più informazioni tra tutti.



## TEST SVM GRAYSCALE

Come test finale per l'SVM sono state fornite al modello le immagini in scala di grigi, sia con il kernel lineare che con il poly.

KERNEL LINEAR	KERNEL POLY
 <p>True label</p> <p>B</p> <p>N</p> <p>B</p> <p>N</p> <p>Predicted label</p>	 <p>True label</p> <p>B</p> <p>N</p> <p>B</p> <p>N</p> <p>Predicted label</p>
<ul style="list-style-type: none"> <li>• Accuracy: 75.80%</li> <li>• Precision: 76.89%</li> <li>• Sensitivity: 73.76%</li> <li>• Specificity: 77.83%</li> <li>• F1-score: 75.29%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy: 78.81%</li> <li>• Precision: 84.73%</li> <li>• Sensitivity: 70.3%</li> <li>• Specificity: 87.33%</li> <li>• F1-score: 76.84%</li> </ul>
Parametri: - C = 0.1	Parametri - C = 0.1 - Gamma = 0.01

I test con le immagini ridotte a scala di grigi non mostrano un miglioramento rispetto alle immagini originali, anzi i test effettuati sui colori riescono a fornirci risultati migliori, soprattutto quelli sul canale blu.

Questa strada non sembra essere la migliore per il support vector machine.

## TEST SVM FILTRI

I filtri utilizzati in questo task sono stati spiegati nel capitolo 3. Anche in questo sono state utilizzate 12000 immagini 32x32, 6000 biodegradabili e 6000 non biodegradabili.

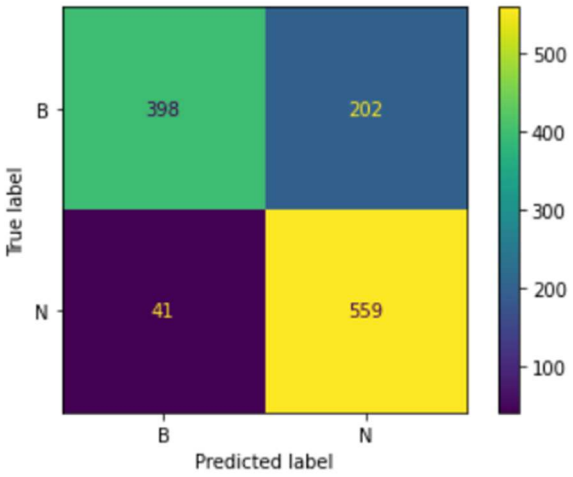
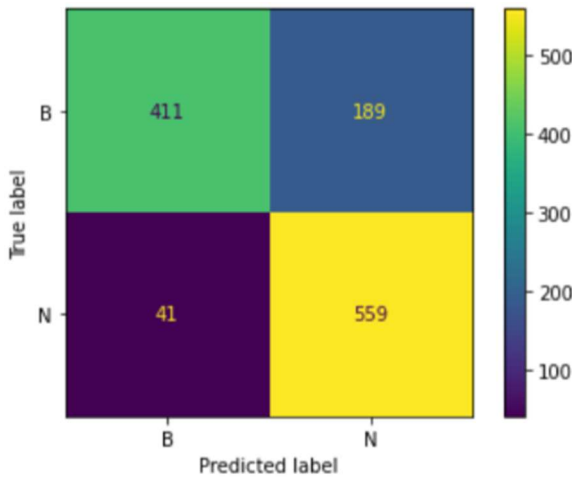
DFT	SOBEL	ENTROPY																											
<p>Confusion matrix for DFT filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>1555</td> <td>245</td> </tr> <tr> <td>N</td> <td>265</td> <td>1535</td> </tr> </table>	True label \ Predicted label	B	N	B	1555	245	N	265	1535	<p>Confusion matrix for SOBEL filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2444</td> <td>556</td> </tr> <tr> <td>N</td> <td>683</td> <td>2317</td> </tr> </table>	True label \ Predicted label	B	N	B	2444	556	N	683	2317	<p>Confusion matrix for ENTROPY filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2619</td> <td>381</td> </tr> <tr> <td>N</td> <td>996</td> <td>2004</td> </tr> </table>	True label \ Predicted label	B	N	B	2619	381	N	996	2004
True label \ Predicted label	B	N																											
B	1555	245																											
N	265	1535																											
True label \ Predicted label	B	N																											
B	2444	556																											
N	683	2317																											
True label \ Predicted label	B	N																											
B	2619	381																											
N	996	2004																											
<ul style="list-style-type: none"> <li>• Accuracy = 85.83%</li> <li>• Precision = 86.23%</li> <li>• Sensitivity = 85.27%</li> <li>• Specificity = 86.38%</li> <li>• F1-Score = 85.75%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 79.35%</li> <li>• Precision = 80.64%</li> <li>• Sensitivity = 77.23%</li> <li>• Specificity = 81.46%</li> <li>• F-score = 78.90%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 77.05%</li> <li>• Precision = 84.02%</li> <li>• Sensitivity = 66.80%</li> <li>• Specificity = 87.3%</li> <li>• F-score = 74.42%</li> </ul>																											
Parametri: - C = 0.1 - Kernel = 'linear'	Parametri: - C = 0.1 - Kernel = 'rbf'	Parametri: - C = 0.1 - Kernel = 'rbf'																											

In questo test è stato ricapitolato il risultato avuto fornendo al support vector machine le immagini a cui sono stati applicati i vari filtri. È evidente che il filtro migliore è la discrete fourier transform che applicata alle immagini riesce a dare un risultato entusiasmante, ovvero il parametro migliore per i modelli trainati con queste immagini è il kernel linear, segno che i dati diventano linearmente separabili. Per quanto riguarda gli altri modelli trainati con dati filtrati dal sobel e l'entropia riescono a raggiungere buoni risultati ma non sorprendono.

## 4.3 K- NEAREST NEIGHBOR

### TEST K- NEAREST NEIGHBOR 01

Test effettuato su 6000 immagini di size 32x32:

	
<ul style="list-style-type: none"><li>• Accuracy = 79.75%</li><li>• Precision = 73.45%</li><li>• Sensitivity = 93.16%</li><li>• Specificity = 66.33%</li><li>• F1-Score = 82.14%</li></ul>	<ul style="list-style-type: none"><li>• Accuracy = 80.83%</li><li>• Precision = 74.73%</li><li>• Sensitivity = 93.16%</li><li>• Specificity = 68.50 %</li><li>• F1-Score = 82.93%</li></ul>
I parametri utilizzati sono i seguenti: <ul style="list-style-type: none"><li>- Metric = 'Euclidian'</li><li>- N_neighbours = 13</li><li>- Weights = 'distance'</li></ul>	I parametri utilizzati sono i seguenti: <ul style="list-style-type: none"><li>- Metric = 'Manhattan'</li><li>- N_neighbours = 5</li><li>- Weights = 'distance'</li></ul>

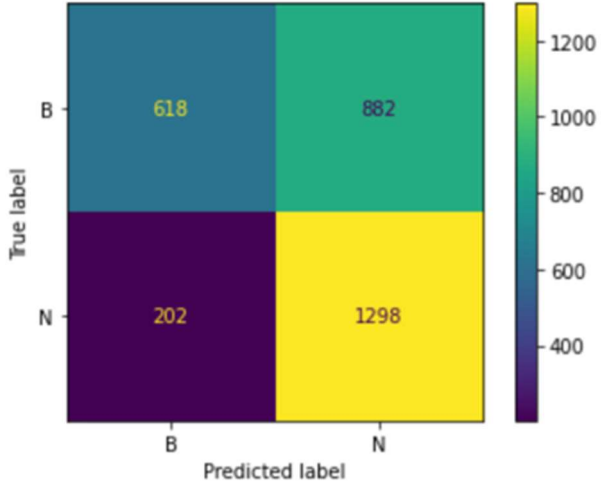
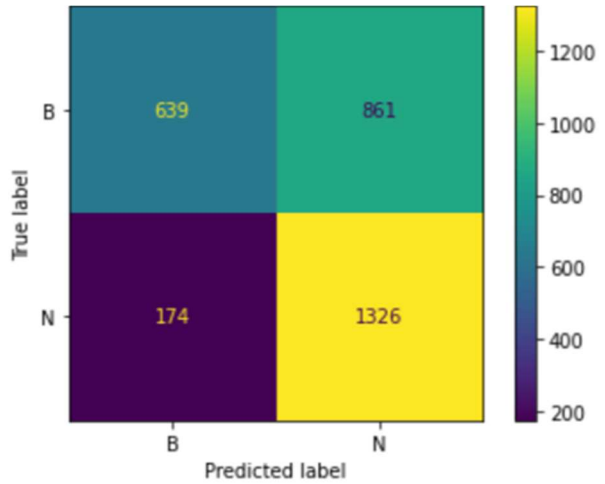
Il K- nearest neighbor applicato sulle immagini originali fornisce dei risultati leggermente peggiori di quanto fatto sulle stesse immagini dagli altri due modelli.

La cosa interessante da notare è che la metrica manhattan è leggermente migliore di quella euclidea, nonostante il parametro n neighbours in questo modello sia minore.

## TEST K- NEAREST NEIGHBOR ANALISI RGB

In questo esperimento per il K- nearest neighbor sono state analizzate le due distanze, euclidea e di Manhattan. Gli esperimenti per tutti e 3 i canali sono stati realizzati su 12000 immagini di size 32x32:

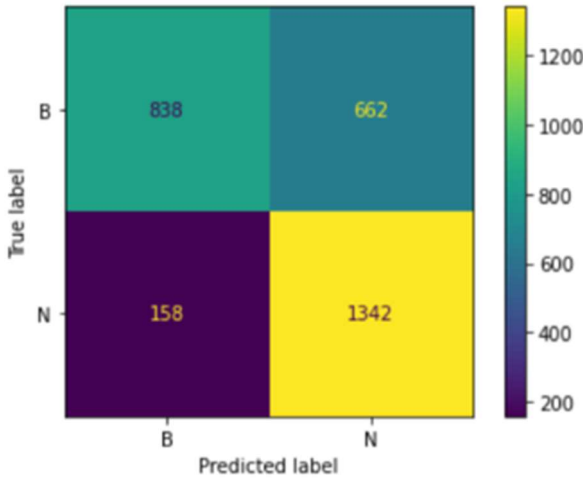
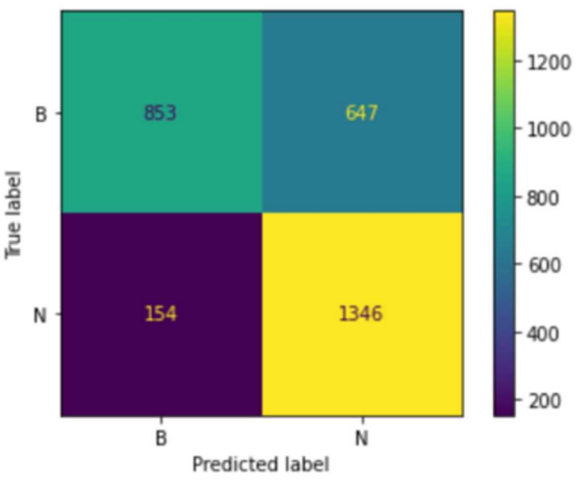
### TEST CANALE ROSSO

 <table><tr><th></th><th>Predicted B</th><th>Predicted N</th></tr><tr><th>True B</th><td>618</td><td>882</td></tr><tr><th>True N</th><td>202</td><td>1298</td></tr></table>		Predicted B	Predicted N	True B	618	882	True N	202	1298	 <table><tr><th></th><th>Predicted B</th><th>Predicted N</th></tr><tr><th>True B</th><td>639</td><td>861</td></tr><tr><th>True N</th><td>174</td><td>1326</td></tr></table>		Predicted B	Predicted N	True B	639	861	True N	174	1326
	Predicted B	Predicted N																	
True B	618	882																	
True N	202	1298																	
	Predicted B	Predicted N																	
True B	639	861																	
True N	174	1326																	
<ul style="list-style-type: none"><li>• Accuracy = 63.86%</li><li>• Precision = 59.54%</li><li>• Sensitivity = 86.53%</li><li>• Specificity = 41.19%</li><li>• F1-Score = 70.54%</li></ul>	<ul style="list-style-type: none"><li>• Accuracy = 65.50%</li><li>• Precision = 60.63%</li><li>• Sensitivity = 88.40%</li><li>• Specificity = 42.60 %</li><li>• F1-Score = 71.92%</li></ul>																		
<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"><li>- Metric = 'Euclidian'</li><li>- N_neighbors = 5</li><li>- Weights = 'distance'</li></ul>	<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"><li>- Metric = 'Manhattan'</li><li>- N_neighbors = 5</li><li>- Weights = 'distance'</li></ul>																		

Da questo test possiamo osservare come il K- nearest neighbor inizia a perdere colpi arrivando addirittura al 63.86 con la metrica settata a euclidian.

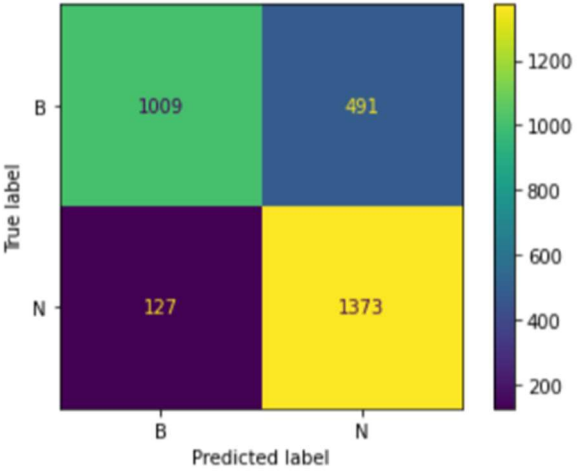
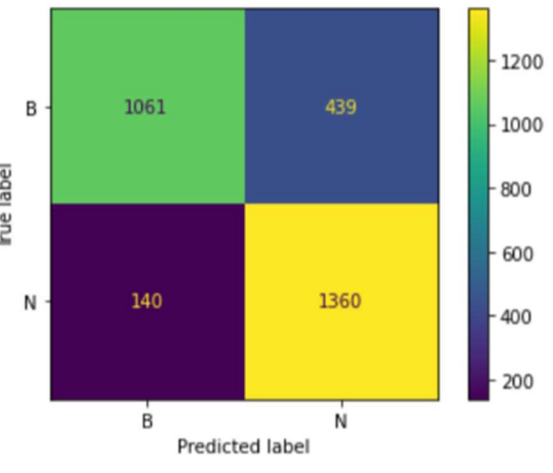
La specificity di questo test è veramente bassa in entrambi i modelli, quindi predire le immagini non biodegradabili risulta abbastanza difficile.

## TEST CANALE VERDE

	
<ul style="list-style-type: none"> <li>• Accuracy = 72.66%</li> <li>• Precision = 66.96%</li> <li>• Sensitivity = 89.46%</li> <li>• Specificity = 55.86%</li> <li>• F1-Score = 76.59%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 73.30%</li> <li>• Precision = 67.53%</li> <li>• Sensitivity = 89.73%</li> <li>• Specificity = 56.86 %</li> <li>• F1-Score = 77.06%</li> </ul>
<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Metric = 'Euclidian'</li> <li>- N_neighbors = 9</li> <li>- Weights = 'distance'</li> </ul>	<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Metric = 'Manhattan'</li> <li>- N_neighbors = 13</li> <li>- Weights = 'distance'</li> </ul>

Il canale verde riesce a risollevare il punteggio dell'accuracy di circa il 7 8% nei vari modelli testati. Ciò è sintomo di un incremento abbastanza sostanzioso delle informazioni che ci fornisce questo canale, in linea con quello successo con i test per il svm.

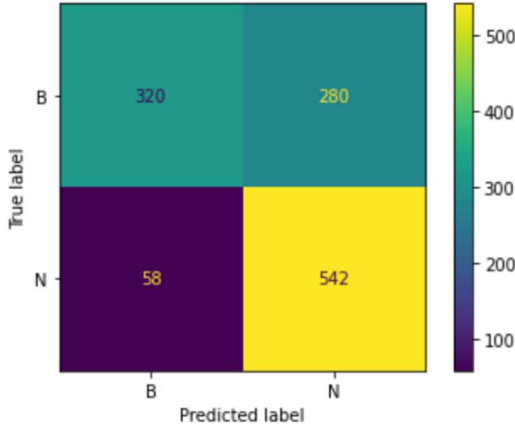
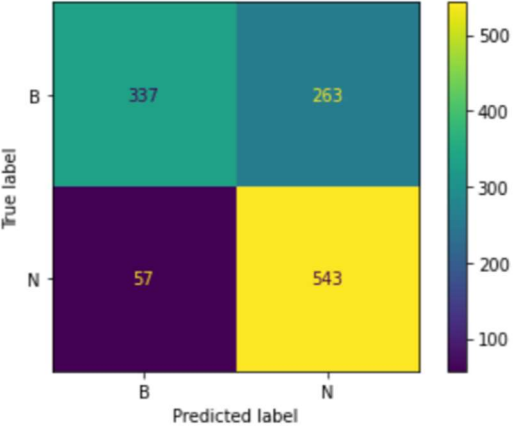
## TEST CANALE BLU

 <p>True label</p> <p>B</p> <p>N</p> <p>B</p> <p>N</p> <p>Predicted label</p>	 <p>True label</p> <p>B</p> <p>N</p> <p>B</p> <p>N</p> <p>Predicted label</p>
<ul style="list-style-type: none"> <li>• Accuracy = 79.40%</li> <li>• Precision = 73.65%</li> <li>• Sensitivity = 91.53%</li> <li>• Specificity = 67.26%</li> <li>• F1-Score = 81.62%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 80.70%</li> <li>• Precision = 75.59%</li> <li>• Sensitivity = 90.66%</li> <li>• Specificity = 70.73 %</li> <li>• F1-Score = 82.44%</li> </ul>
<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Metric = 'Euclidian'</li> <li>- N_neighbors = 13</li> <li>- Weights = 'distance'</li> </ul>	<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Metric = 'Manhattan'</li> <li>- N_neighbors = 7</li> <li>- Weights = 'distance'</li> </ul>

Con questo canale si raggiungono i risultati ottenuti dai vari modelli del K- nearest neighbor visti precedentemente dove sono stati utilizzati tutti i canali, con uno scarto del solo 0.13%, il che è un risultato abbastanza soddisfacente.

## TEST K- NEAREST NEIGHBOR GRAYSCALE

Anche in questo caso il test è effettuato su 12 000 immagini 32x32.

 <p>True label</p> <p>B</p> <p>N</p> <p>B</p> <p>N</p> <p>Predicted label</p>	 <p>True label</p> <p>B</p> <p>N</p> <p>B</p> <p>N</p> <p>Predicted label</p>
<ul style="list-style-type: none"> <li>• Accuracy = 71.83%</li> <li>• Precision = 65.93%</li> <li>• Sensitivity = 90.33%</li> <li>• Specificity = 53.33%</li> <li>• F1-Score = 76.23%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 73.33%</li> <li>• Precision = 67.36%</li> <li>• Sensitivity = 90.55%</li> <li>• Specificity = 56.16 %</li> <li>• F1-Score = 77.24%</li> </ul>
<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Metric = 'Euclidian'</li> <li>- N_nieghbors = 26</li> <li>- Weights = 'distance'</li> </ul>	<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Metric = 'Manhattan'</li> <li>- N_nieghbors = 26</li> <li>- Weights = 'distance'</li> </ul>

I risultati ci mostrano che con entrambi i tipi di distanza i risultati non sono così entusiasmanti, tuttavia quello che possiamo notare la sensibilità è molto elevata, in contrapposizione con la specificità che è un po' bassa rispetto alla media dei risultati ottenuti.

## TEST K- NEAREST NEIGHBOR FILTRI

I filtri utilizzati in questo task sono stati spiegati nel capitolo 3. Anche in questo sono state utilizzate 12000 immagini 32x32, 6000 biodegradabili e 6000 non biodegradabili.

DFT	SOBEL	ENTROPY																											
<p>Confusion matrix for DFT filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2855</td> <td>145</td> </tr> <tr> <td>N</td> <td>856</td> <td>2144</td> </tr> </table>	True label \ Predicted label	B	N	B	2855	145	N	856	2144	<p>Confusion matrix for SOBEL filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2497</td> <td>503</td> </tr> <tr> <td>N</td> <td>660</td> <td>2340</td> </tr> </table>	True label \ Predicted label	B	N	B	2497	503	N	660	2340	<p>Confusion matrix for ENTROPY filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2277</td> <td>723</td> </tr> <tr> <td>N</td> <td>826</td> <td>2174</td> </tr> </table>	True label \ Predicted label	B	N	B	2277	723	N	826	2174
True label \ Predicted label	B	N																											
B	2855	145																											
N	856	2144																											
True label \ Predicted label	B	N																											
B	2497	503																											
N	660	2340																											
True label \ Predicted label	B	N																											
B	2277	723																											
N	826	2174																											
<ul style="list-style-type: none"> <li>• Accuracy = 83.31%</li> <li>• Precision = 93.66%</li> <li>• Sensitivity = 71.46%</li> <li>• Specificity = 95.16%</li> <li>• F1-Score = 81.07%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 76.50%</li> <li>• Precision = 84.38%</li> <li>• Sensitivity = 65.03%</li> <li>• Specificity = 87.96%</li> <li>• F-score = 73.45%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 74.18%</li> <li>• Precision = 75.04%</li> <li>• Sensitivity = 72.46%</li> <li>• Specificity = 75.90%</li> <li>• F-score = 73.73%</li> </ul>																											
Parametri: <ul style="list-style-type: none"> <li>- Metric = Manhattan</li> <li>- N_neighbors = 5</li> <li>- Weights = distance</li> </ul>	Parametri: <ul style="list-style-type: none"> <li>- Metric = Manhattan</li> <li>- N_neighbors = 13</li> <li>- Weights = distance</li> </ul>	Parametri <ul style="list-style-type: none"> <li>- Metric = Manhattan</li> <li>- N_neighbors = 13</li> <li>- Weights = distance</li> </ul>																											

In questo test sono stati presi i parametri dei modelli migliori, trainati con le immagini a cui sono stati applicati i vari filtri. Come possiamo osservare tutti danno dei risultati migliori con il parametro distance settato a manhattan.

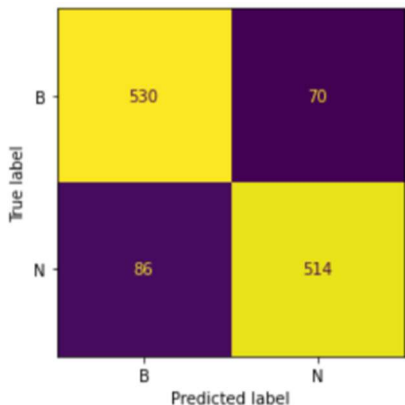
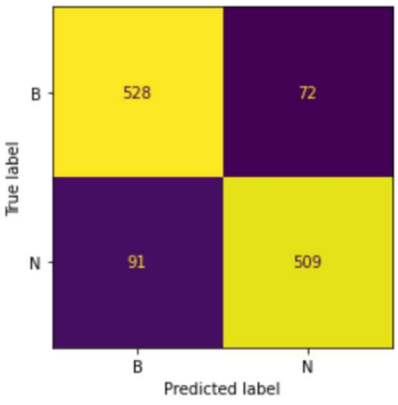
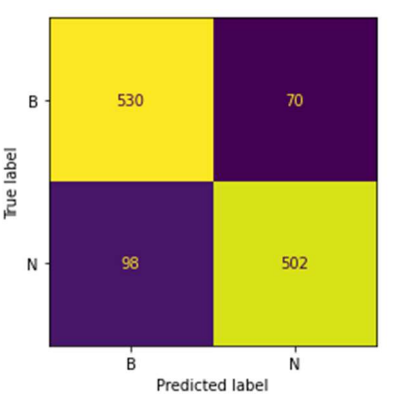
Ovviamente anche questa volta il miglior filtro si rivela essere la discrete transform fourier, dove è interessante notare la specificity supera il 95% e la precisione il 93%, cioè la percentuale delle immagini biodegradabili rivelatesi poi realmente tale è alta, mentre la specificity spiega che riesce a predire bene le immagini non biodegradabili, ma per qualche ragione fa difficoltà a riconoscere le immagini biodegradabili.



## 4.4 RANDOM FOREST

### TEST 01 RANDOM FOREST

Il random forest è risultato il modello più difficile da ottimizzare. In questo primo test sono stati presi in considerazione i parametri più importanti max features e il numero di decision trees, n\_estimators. Come negli altri test sono state prese in considerazione 6000 immagini 32x32:

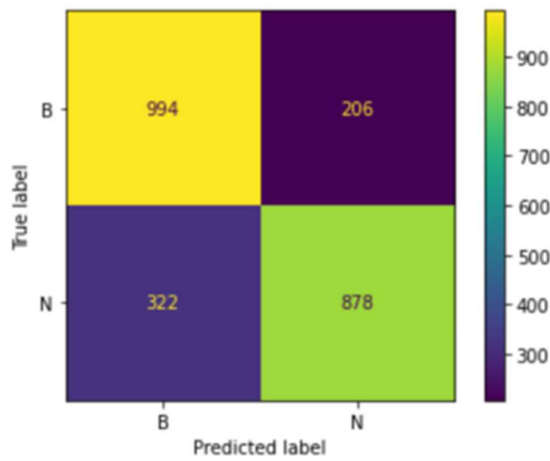
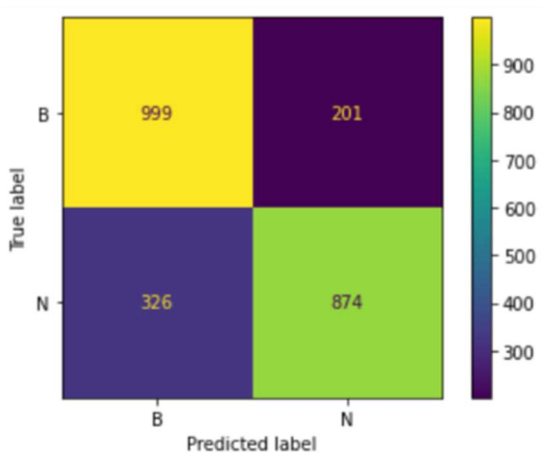
		
<ul style="list-style-type: none"> <li>• Accuracy = 87.00%</li> <li>• Precision = 88.01%</li> <li>• Sensitivity = 85.66%</li> <li>• Specificity = 88.33%</li> <li>• F1-Score = 86.82%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 86.41%</li> <li>• Precision = 87.60%</li> <li>• Sensitivity = 84.83%</li> <li>• Specificity = 88.00%</li> <li>• F1-Score = 86.19%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 86.00%</li> <li>• Precision = 87.76%</li> <li>• Sensitivity = 83.66%</li> <li>• Specificity = 88.33%</li> <li>• F1-Score = 85.65%</li> </ul>
Parametri: <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max_depth = 70</li> <li>- n_estimators = 300</li> <li>- max_features = None</li> </ul>	Parametri: <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max_depth = 50</li> <li>- n_estimators = 300</li> <li>- max_features = Sqrt</li> </ul>	Parametri: <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max_depth = 50</li> <li>- n_estimators = 300</li> <li>- max_features = log2</li> </ul>

Il test in questione vuole mostrare come l'accuratezza va diminuendo al diminuire delle feature prese in considerazione, tuttavia il gap tra il modello migliore e il peggiore è di solo un punto percentuale.

## TEST RANDOM FOREST ANALISI RGB

### TEST CANALE ROSSO

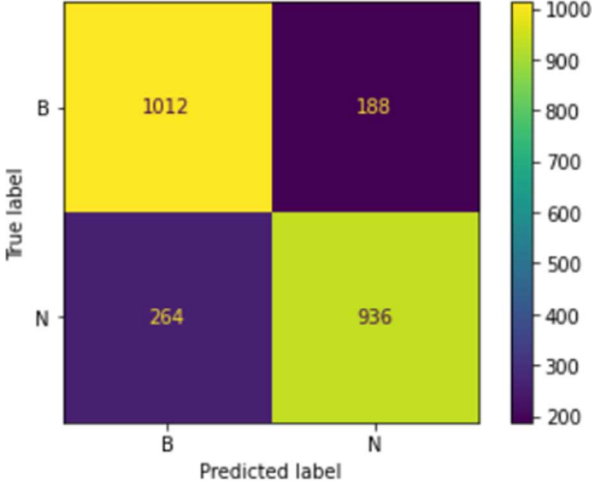
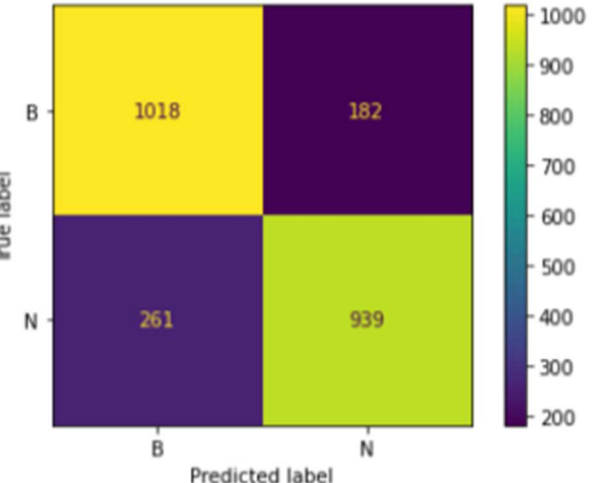
Test effettuato con 12000 immagini 32x32.

 <table><tr><th></th><th>Predicted label B</th><th>Predicted label N</th></tr><tr><th>True label B</th><td>994</td><td>206</td></tr><tr><th>True label N</th><td>322</td><td>878</td></tr></table>		Predicted label B	Predicted label N	True label B	994	206	True label N	322	878	 <table><tr><th></th><th>Predicted label B</th><th>Predicted label N</th></tr><tr><th>True label B</th><td>999</td><td>201</td></tr><tr><th>True label N</th><td>326</td><td>874</td></tr></table>		Predicted label B	Predicted label N	True label B	999	201	True label N	326	874
	Predicted label B	Predicted label N																	
True label B	994	206																	
True label N	322	878																	
	Predicted label B	Predicted label N																	
True label B	999	201																	
True label N	326	874																	
<ul style="list-style-type: none"><li>• Accuracy = 78.00%</li><li>• Precision = 80.99%</li><li>• Sensitivity = 73.16%</li><li>• Specificity = 82.83%</li><li>• F1-Score = 76.88%</li></ul>	<ul style="list-style-type: none"><li>• Accuracy = 78.04%</li><li>• Precision = 81.30%</li><li>• Sensitivity = 72.83%</li><li>• Specificity = 83.25 %</li><li>• F1-Score = 76.83%</li></ul>																		
I parametri utilizzati sono i seguenti: <ul style="list-style-type: none"><li>- Bootstrap = True</li><li>- Max depth = 30</li><li>- N estimators = 100</li></ul>	I parametri utilizzati sono i seguenti: <ul style="list-style-type: none"><li>- Bootstrap = True</li><li>- Max depth = 70</li><li>- N estimators = 300</li></ul>																		

Il pattern si ripete. Il canale rosso è ancora il canale peggiore, ma nonostante ciò raggiunge un discreto risultato con il random forest, simbolo di un modello abbastanza robusto. Possiamo notare che da 100 estimators a 300 estimators con la profondità aumentata, l'accuratezza è del solo dello 0.4% in più.

## TEST CANALE VERDE

Test effettuato con 12 000 immagini 32x32.

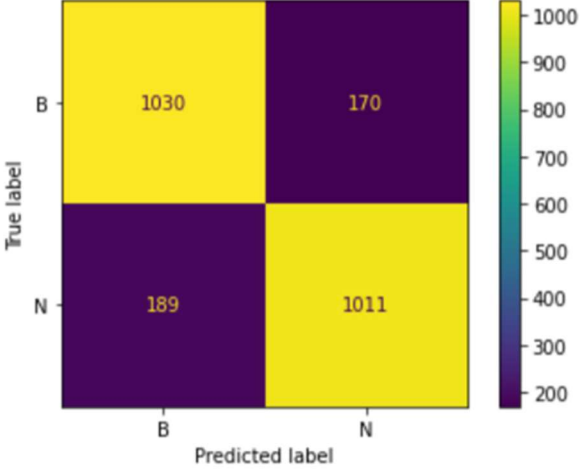
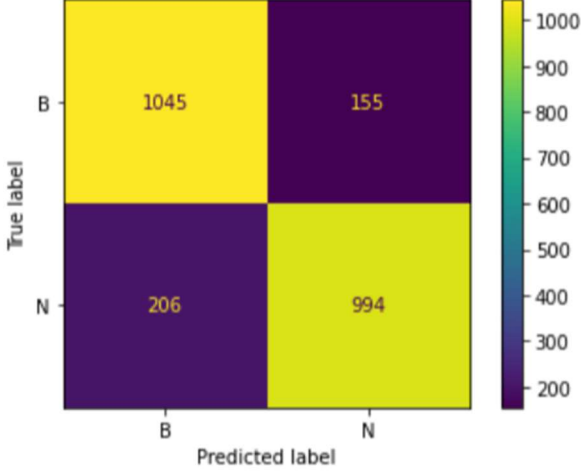
	
<ul style="list-style-type: none"> <li>• Accuracy = 81.16%</li> <li>• Precision = 83.27%</li> <li>• Sensitivity = 78.00%</li> <li>• Specificity = 84.33%</li> <li>• F1-Score = 80.55%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 81.54%</li> <li>• Precision = 83.76%</li> <li>• Sensitivity = 78.25%</li> <li>• Specificity = 84.83 %</li> <li>• F1-Score = 80.91%</li> </ul>
<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max depth = 30</li> <li>- N estimators = 100</li> </ul>	<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max depth = 70</li> <li>- N estimators = 300</li> </ul>

Questo test conferma ancora una volta che il canale verde è migliore di quello rosso.

Abbiamo qui i due modelli migliori che hanno fornito risultati abbastanza simili anche con parametri abbastanza diversi.

## TEST CANALE BLU

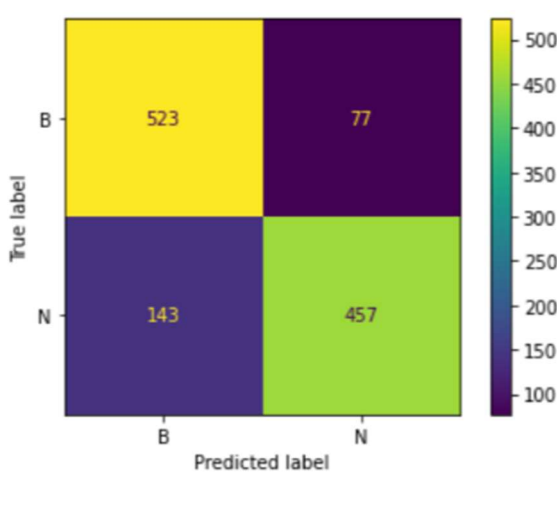
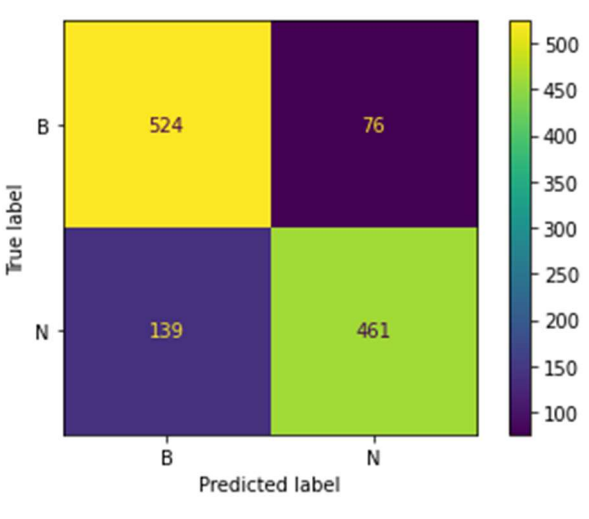
Test effettuato con 12 000 immagini 32x32.

	
<ul style="list-style-type: none"> <li>• Accuracy = 85.04%</li> <li>• Precision = 85.60%</li> <li>• Sensitivity = 84.25%</li> <li>• Specificity = 85.83%</li> <li>• F1-Score = 84.92%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 84.95%</li> <li>• Precision = 86.51%</li> <li>• Sensitivity = 82.83%</li> <li>• Specificity = 87.08 %</li> <li>• F1-Score = 84.63%</li> </ul>
<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max depth = 30</li> <li>- N estimators = 100</li> </ul>	<p>I parametri utilizzati sono i seguenti:</p> <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max depth = 70</li> <li>- N estimators = 300</li> </ul>

Come ci si aspettava anche con il random forest questo canale fornisce risultati migliori, tuttavia è interessante notare che il trend si inverte per quanto riguarda i parametri. In questo caso a fare leggermente meglio è il modello con meno estimators.

## TEST RANDOM FOREST GRAYSCALE

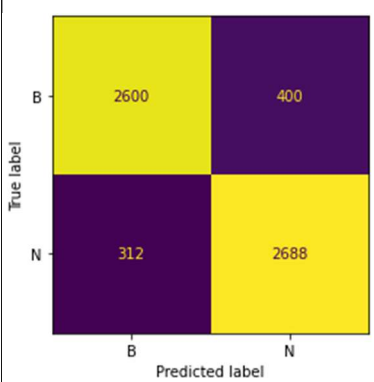
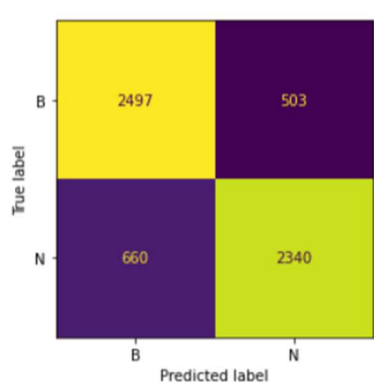
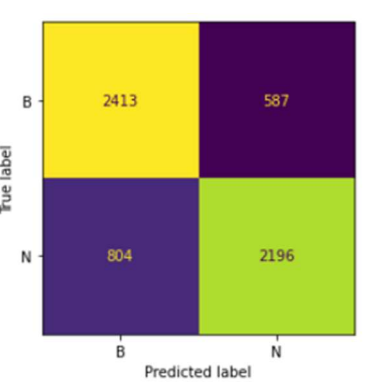
Test effettuato con 12 000 immagini 32x32.

 <p>Confusion matrix for Random Forest Grayscale (300 estimators):</p> <table><tr><th>True label \ Predicted label</th><th>B</th><th>N</th></tr><tr><th>B</th><td>523</td><td>77</td></tr><tr><th>N</th><td>143</td><td>457</td></tr></table>	True label \ Predicted label	B	N	B	523	77	N	143	457	 <p>Confusion matrix for Random Forest Grayscale (100 estimators):</p> <table><tr><th>True label \ Predicted label</th><th>B</th><th>N</th></tr><tr><th>B</th><td>524</td><td>76</td></tr><tr><th>N</th><td>139</td><td>461</td></tr></table>	True label \ Predicted label	B	N	B	524	76	N	139	461
True label \ Predicted label	B	N																	
B	523	77																	
N	143	457																	
True label \ Predicted label	B	N																	
B	524	76																	
N	139	461																	
<ul style="list-style-type: none"><li>• Accuracy = 81.66%</li><li>• Precision = 85.58%</li><li>• Sensitivity = 76.16%</li><li>• Specificity = 87.16%</li><li>• F1-Score = 80.59%</li></ul>	<ul style="list-style-type: none"><li>• Accuracy = 82.08%</li><li>• Precision = 85.84%</li><li>• Sensitivity = 76.83%</li><li>• Specificity = 87.33 %</li><li>• F1-Score = 81.09%</li></ul>																		
I parametri utilizzati sono i seguenti: <ul style="list-style-type: none"><li>- Bootstrap = True</li><li>- Max depth = 50</li><li>- N_estimators = 300</li></ul>	I parametri utilizzati sono i seguenti: <ul style="list-style-type: none"><li>- Bootstrap = True</li><li>- Max depth = 50</li><li>- N_estimators = 100</li></ul>																		

Questo test conferma come la scala di grigi non riesca a fornire un valido sostituto alle altre soluzioni proposte in questa analisi. Infatti nonostante i risultati siano buoni sono ben lontani da quelli ottenuti dai modelli trainati su tutti i canali, anche il modello con solo il canale blue riesce a fare meglio.

## TEST RANDOM FOREST FILTRI

I filtri utilizzati in questo task sono stati spiegati nel capitolo 3. Anche in questo sono state utilizzate 12000 immagini 32x32, 6000 biodegradabili e 6000 non biodegradabili.

DFT	SOBEL	ENTROPY																											
 <p>Confusion matrix for DFT filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2600</td> <td>400</td> </tr> <tr> <td>N</td> <td>312</td> <td>2688</td> </tr> </table>	True label \ Predicted label	B	N	B	2600	400	N	312	2688	 <p>Confusion matrix for SOBEL filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2497</td> <td>503</td> </tr> <tr> <td>N</td> <td>660</td> <td>2340</td> </tr> </table>	True label \ Predicted label	B	N	B	2497	503	N	660	2340	 <p>Confusion matrix for ENTROPY filter:</p> <table border="1"> <tr> <td>True label \ Predicted label</td> <td>B</td> <td>N</td> </tr> <tr> <td>B</td> <td>2413</td> <td>587</td> </tr> <tr> <td>N</td> <td>804</td> <td>2196</td> </tr> </table>	True label \ Predicted label	B	N	B	2413	587	N	804	2196
True label \ Predicted label	B	N																											
B	2600	400																											
N	312	2688																											
True label \ Predicted label	B	N																											
B	2497	503																											
N	660	2340																											
True label \ Predicted label	B	N																											
B	2413	587																											
N	804	2196																											
<ul style="list-style-type: none"> <li>• Accuracy = 88.13%</li> <li>• Precision = 87.04%</li> <li>• Sensitivity = 89.00%</li> <li>• Specificity = 86.66%</li> <li>• F1-Score = 88.30%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 80.61%</li> <li>• Precision = 82.30%</li> <li>• Sensitivity = 78.00%</li> <li>• Specificity = 83.22%</li> <li>• F-score = 80.09%</li> </ul>	<ul style="list-style-type: none"> <li>• Accuracy = 76.81%</li> <li>• Precision = 78.90%</li> <li>• Sensitivity = 73.20%</li> <li>• Specificity = 80.43%</li> <li>• F-score = 75.94%</li> </ul>																											
Parametri: <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max_depth = 70</li> <li>- n_estimators = 300</li> </ul>	Parametri: <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max_depth = 70</li> <li>- n_estimators = 300</li> </ul>	Parametri <ul style="list-style-type: none"> <li>- Bootstrap = True</li> <li>- Max_depth = 50</li> <li>- n_estimators = 400</li> </ul>																											

Il random forest dà il suo meglio con l'analisi dei filtri. Questi test confermano ulteriormente la poca utilità dei filtri sobel ed entropy, e affermano la superiorità del filtro DFT che riesce a dare il meglio di superando l'88% dove fin'ora nessun modello con nessuna combinazione di parametri è riuscito.

## 4.5 TABELLA RIASSUNTIVA

Questa tabella riassume i vari risultati ottenuti nei test. Si evidenzia in verde il migliore di tutti, in grassetto i migliori di quella categoria.

Metriche	Modelli	ACCURACY	PRECISION	SENSITIVITY	SPECIFICITY	F-SCORE
immagini originali	SVM	<b>88.00%</b>	89.90%	85.73%	90.26%	87.72%
	KNN	80.83%	74.73%	93.16%	68.50%	82.93%
	RANDOM FOREST	87.00%	88.01%	85.66%	88.33%	86.82%
Analisi canale rosso	SVM	68.77%	68.61%	69.22%	68.33%	68.91%
	KNN	65.50%	60.63%	88.40%	42.60%	71.92%
	RANDOM FOREST	<b>78.04%</b>	81.30%	72.83%	83.25%	76.83%
Analisi canale verde	SVM	76.94%	78.29%	74.55%	79.33%	76.38%
	KNN	73.30%	67.53%	89.73%	56.86%	77.06%
	RANDOM FOREST	<b>81.54%</b>	83.76%	78.25%	84.83%	80.91%
Analisi canale blu	SVM	81.75%	82.56%	80.50%	83.00%	81.51%
	KNN	80.70%	75.59%	90.66%	70.73%	82.44%
	RANDOM FOREST	<b>85.04%</b>	85.60%	84.25%	85.83%	84.92%
Analisi Scala di grigi	SVM	78.81%	84.73%	70.3%	87.33%	76.84%
	KNN	73.33%	67.36%	90.55%	56.16%	77.24%
	RANDOM FOREST	<b>82.08%</b>	85.84%	76.83%	87.33%	81.09%
Dft	SVM	85.83%	86.23%	85.27%	86.38%	85.75%
	KNN	83.31%	93.66%	71.46%	95.16%	81.07%
	RANDOM FOREST	<b>88.13%</b>	87.04%	89.00%	86.66%	88.30%
Edge Detection	SVM	79.35%	80.64%	77.23%	81.46%	78.90%
	KNN	76.50%	84.38%	65.03%	87.96%	73.45%
	RANDOM FOREST	<b>80.61%</b>	82.30%	78.00%	83.22%	80.09%
Entropy	SVM	<b>77.05%</b>	84.02%	66.80%	87.30%	74.42%
	KNN	74.18%	75.04%	72.46%	75.90%	73.73%
	RANDOM FOREST	76.81%	78.90%	73.20%	80.43%	75.94%

## 5. CONCLUSIONE E LAVORI FUTURI

In questo progetto di tesi sono state utilizzate tecniche di computer vision e machine learning per creare un classificatore binario in grado di saper distinguere le due categorie di dati del dataset “Non and Biodegradable Material”. Inizialmente, si è cercato di ridurre il numero di feature delle immagini con l’analisi dello spazio RGB, e da questo si è scoperto che il canale blu è quello più informativo, mentre il rosso risulta esserlo meno pur mantenendo una discreta quantità di dettagli. Dall’analisi dello spazio dei colori si è poi passati all’analisi delle immagini in scala di grigi, le quali però non ha fornito dati interessanti dato che non si raggiunge una particolare accuratezza. Si è passati poi all’analisi di alcuni filtri per cercare di ricavare informazioni dalle immagini atte a tale scopo. Questi filtri sono l’entropy, il sobel e la discrete fourier transform. Come risultato dai test, è stato ottenuto che l’unico di essi che aumenta l’accuratezza del modello, e che quindi fornisce delle informazioni utili allo scopo di classificazione, è la discrete fourier transform.

Tali test sono stati accompagnati da un processo che in letteratura è noto come hyperparameter optimization, ovvero per ogni modello si è provato un set di parametri con il fine ultimo di cercare quale combinazione migliori l’accuratezza del modello. Per il support vector machine si sono ottenute le informazioni più interessanti, scoprendo che il dataset con le immagini originali non è linearmente separabile e quindi ha bisogno dei kernel Radial Basis Function o poly, delle funzioni che permettono di mappare le immagini in spazi dimensionali maggiori in modo da cercare di renderle linearmente separabili. Successivamente, con l’analisi nello spazio RGB, il kernel lineare ritorna a dare dei buoni risultati, migliori di rbf e poly. Infine, addestrando questo modello con le immagini filtrate, si scopre che l’accuratezza migliore è fornita dal kernel lineare sulle immagini a cui è applicata la discrete fourier transform.

Per quanto riguarda il K- nearest neighbor, in tutti i test, sia per le immagini originali, che per l’analisi RGB, che per l’analisi dei filtri, i vari risultati hanno evidenziato la differenza tra due valori del parametro distanza, che determinano il modo in cui viene calcolata la



distanza tra il punto da classificare e tutti gli altri nel dataset, favorendo manhattan al posto di euclidian, ma non raggiungendo gli altri modelli come accuratezza.

Il random forest, invece, è stato il modello più difficile da ottimizzare, concordando con ciò che è noto in letteratura. Infatti, anche cambiando i vari parametri, l'accuratezza dei diversi modelli ha una bassa oscillazione, variando di pochi punti percentuali. Ma nonostante ciò in media ha un punteggio molto elevato.

In conclusione, il random forest e il support vector machine risultano essere gli algoritmi migliori. Il punteggio più alto è stato effettuato dal random forest addestrato sul dataset delle immagini a cui è stata applicata la discrete fourier transform ottenendo l'88.13% di accuratezza.

Concludendo, in generale in letteratura si ottengono per il task di classificazione risultati migliori utilizzando tecniche di deep learning, l'idea futura infatti, è quella di esplorare queste tecniche per risolvere il compito di distinguere da rifiuti e prodotti non biodegradabili. L'intenzione è quella di costruire un classificatore multiclasse che riesca a distinguere più tipi di rifiuti, andando sempre di più verso una tecnologia di supporto e sempre più orientata alla salvaguardia e alla responsabilizzazione sulle tematiche ambientali.

## BIBLIOGRAFIA

- [1] Evgeniou, Theodoros & Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7\_12.
- [2] K. Taunk, S. De, S. Verma and A. Swetapadma, "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), 2019, pp. 1255-1260, doi: 10.1109/ICCS45141.2019.9065747.
- [3] Breiman, L. Random Forests. *Machine Learning* **45**, 5–32 (2001).  
<https://doi.org/10.1023/A:1010933404324>
- [4] Mahesh, Batta. (2019). Machine Learning Algorithms -A Review. 10.21275/ART20203995.
- [5] Alzubaidi, L., Zhang, J., Humaidi, A.J. *et al.* Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J Big Data* **8**, 53 (2021).  
<https://doi.org/10.1186/s40537-021-00444-8>
- [6] Jolliffe Ian T. and Cadima Jorge 2016Principal component analysis: a review and recent developmentsPhil. Trans. R. Soc. A.3742015020220150202  
<http://doi.org/10.1098/rsta.2015.0202>
- [7] Quinlan, J.R. Induction of decision trees. *Mach Learn* **1**, 81–106 (1986).  
<https://doi.org/10.1007/BF00116251>
- [8] WIREs Data Mining Knowl Discov 2019 arXiv:1804.03515