

# **Titolo tesi**

Sottotitolo

**Giacomo Saccaggi**

Facoltà di  
Scienze statistiche ed economiche



Dipartimento di Statistica  
Università Bicocca  
Sede di Milano  
2019/2020

*Dedica*

# Indice

<b>Introduzione</b>	<b>1</b>
<b>1 Reperimento dei dati</b>	<b>5</b>
1.1 Struttura del lavoro . . . . .	5
1.2 Come differenziare . . . . .	6
1.3 Web Scraping . . . . .	7
1.3.1 Funzionamento . . . . .	7
1.3.2 Implementazione . . . . .	8
1.4 Bot Telegram . . . . .	12
1.4.1 Funzionamento . . . . .	12
1.4.2 Implementazione . . . . .	12
1.5 Il cestino . . . . .	14
1.5.1 Meccanica . . . . .	14
1.5.2 Robotica . . . . .	18
1.5.3 Softwaristica . . . . .	26
<b>2 Addestramento train e modelli</b>	<b>29</b>
2.1 Descrizione tipologie di dati . . . . .	29
2.2 Dati Strutturati . . . . .	29
2.2.1 Spiegazione teorica dei modelli utilizzati . . . . .	29
2.2.2 Applicazione dei modelli . . . . .	29
2.3 Dati non strutturati . . . . .	29
2.3.1 Spiegazione teorica dei modelli utilizzati . . . . .	29
2.3.2 Applicazione dei modelli . . . . .	29

<b>3 Verifica test, confronto dei modelli</b>	<b>30</b>
3.1 Confronto modelli . . . . .	30
3.2 Ensemble Learning . . . . .	30
3.2.1 Spiegazione teorica dei modelli utilizzati . . . . .	30
3.2.2 Applicazione dei modelli . . . . .	30
3.3 Esempi pratici di applicazioni . . . . .	30
3.3.1 Bot di Telegram . . . . .	30
3.3.2 Cestino . . . . .	30
<b>Conclusioni</b>	<b>31</b>
<b>A Codici</b>	<b>32</b>
A.1 Reperimento dei dati . . . . .	32
A.1.1 Codice Web Scraping (R) . . . . .	32
A.1.2 Codice Bot Telegram per salvare i dati (Python) . . . . .	36
A.1.3 Codice Arduino (C++) . . . . .	37
A.1.4 Codice software per salvare i dati (VB.NET) . . . . .	41
A.2 Addestramento dei modelli . . . . .	43
A.2.1 Codice modelli (Python) . . . . .	43
A.3 Applicazioni . . . . .	44
A.3.1 Codice software per differenziare (VB.NET) . . . . .	44
A.3.2 Codice Bot Telegram per differenziare (Python) . . . . .	44

# Introduzione

L'economia italiana è oggi la più performante d'Europa per circolarità di materia, produttività delle risorse e capacità di riciclo. A dimostrarlo sono i numeri del rapporto “L'Economia Circolare in Italia – la filiera del riciclo asse portante di un'economia senza rifiuti” [1]. Il documento, pubblicato nel gennaio del 2019 e curato dall'esperto ambientale Duccio Bianchi di Ambiente Italia, rappresenta il primo vero bilancio sulla “circolarità” nazionale, settore che vale oggi 88 miliardi di fatturato e 22 miliardi di valore aggiunto, ovvero l'1,5 % del PIL.

In questo libro emerge come l'Italia sia attualmente capofila in Europa nei tre indice che l'autore definisce fondamentali per valutare un economia circolare: **tasso di produttività nell'uso delle risorse** (quanti euro di PIL si producono per ogni kg di risorse consumate), **tasso di circolarità della materia nell'economia** (quante materie seconde impieghiamo sul totale dei consumi di materia) e infine il **tasso di riciclo dei rifiuti** (quanti rifiuti, urbani e non urbani, inclusi l'import ed export, avviamo a riciclo internamente). Le performance nazionali risultano non solo superiori alla media UE ma anche alle prestazioni dei principali stati come Germania, Spagna, Regno Unito e Francia.

Il riciclo è un processo di conversione che trasforma i rifiuti in nuovi materiali, oggetti o sostanze del tutto differenti dai rifiuti d'origine. Questo processo porta quattro vantaggi:

- Conserva le risorse: i materiali ricavati vengono convertiti in nuovi prodotti, riducendo la necessità estrarre materie prime dalla Terra, attraverso l'estrazione e la silvicoltura. Il riciclaggio aiuta a conservare importanti materie prime e protegge gli habitat naturali; inoltre può

essere una grande possibilità di evitare importazioni per territori come quello italiano scarso di risorse naturali.

- Consente di risparmiare energia: l'uso di materiali riciclati nel processo di produzione consuma molta meno energia di quella necessaria per la produzione di nuovi prodotti; inoltre, si ottiene un ulteriore risparmio energetico perché è necessaria più energia per estrarre, raffinare, trasportare e elaborare materie prime pronte per l'industria rispetto alla fornitura di materiali pronti per l'industria.
- Aiuta a proteggere l'ambiente: il riciclaggio riduce la necessità di estrazione (estrazione, silvicolture e disboscamento), raffinazione e lavorazione di materie prime che creano un notevole inquinamento dell'aria e dell'acqua. Poiché il riciclaggio consente di risparmiare energia, riduce anche le emissioni di gas serra, il che aiuta a contrastare i cambiamenti climatici.
- Riduce l'accumulo di rifiuti: i materiali riciclabili vengono rielaborati in nuovi prodotti e, di conseguenza, la quantità di rifiuti inviati alle discariche si riduce.

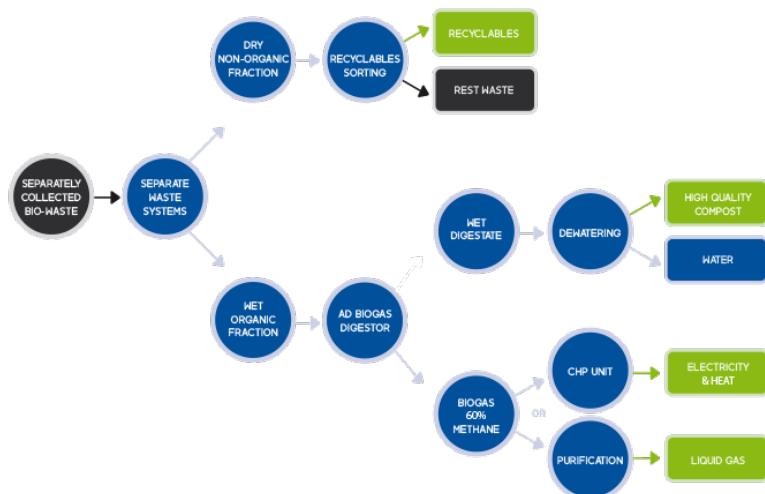


Figura 1: Diversi possibili riutilizzi rifiuti riciclati. (Fonte: separate waste systems)

In quest'ambito subentra il concetto di economia circolare ossia un sistema economico volto ad eliminare gli sprechi e l'uso continuo delle risorse. I sistemi circolari impiegano il riutilizzo, la condivisione, la riparazione, il rinnovo, la rigenerazione e il riciclaggio per creare un sistema a circuito chiuso, riducendo al minimo l'uso di input di risorse e la creazione di rifiuti, inquinamento ed emissioni di carbonio. L'economia circolare mira a mantenere i prodotti, le attrezzature e le infrastrutture in uso più a lungo rispetto invece alla classica economia lineare che si basa sull'utilizzo di risorse sempre nuove e l'eliminazione di quelle vecchie.



Figura 2: Economia circolare. (Fonte: Ecosport economia circolare)

Tra le tante eccellenze italiane nell'ambito del riciclo una delle più importanti riguarda il legno che, come viene evidenziato nell'articolo del giornale Il Sole 24 Ore da Giovanna Mancini [2], in poco più di 20 anni il sistema del recupero e del riciclo del legno ha creato una nuova economia che ha prodotto

risultati importanti sia in termini ambientali, sia per la capacità di creare sviluppo e occupazione. L'impatto economico sulla produzione nazionale delle attività della filiera del recupero del legno post consumo è stimabile, secondo il rapporto del politecnico, in circa 1,4 miliardi di euro, mentre il contributo sull'occupazione è di quasi seimila posti di lavoro complessivamente sostenuti in Italia.

In un industria che sta migrando verso materie prime riciclate la raccolta differenziata, ossia il processo mediante il quale i rifiuti vengono separati in diversi elementi, ricopre un importanza cruciale in quanto più essa viene effettuata con cura e controllo e più il prodotto finale sarà "puro". Questa procedura è essenziale affinché le aziende che si occupano di riciclo possano operare nel settore.

L'obbiettivo di questo lavoro vuole essere quello di affrontare il problema della differenziata da un punto di vista più analitico e proporre alcune idee e soluzioni di come il Data Science possa essere d'aiuto nel riuscire a migliorare lo sfruttamento delle risorse. La dinamica della raccolta differenziata si declina in un problema di classificazione; durante il lavoro si andranno a sfruttare sia dati strutturati che dati non strutturati utilizzando algoritmi di Deep Learning per riuscire a classificare in modo accurato i differenti tipi di rifiuti.

# Capitolo 1

## Reperimento dei dati

### 1.1 Struttura del lavoro

Il riciclo può essere una risorsa molto importante che può non solo garantire un economia più eco sostenibile ma può essere una grande risorsa per l'occupazione e per evitare di importare materie prime.

Per affrontare questo problema si è deciso di strutturare il lavoro in quattro fasi: reperimento dei dati, elaborazione, previsione e infine scelta del modello. Per il carattere applicativo del progetto durante tutto il lavoro si andranno ad approfondire la teoria dei modelli e degli strumenti usati andando ad evidenziare alcune implementazioni ritenute utili ai fini della comprensione, per le altre si lasciano i codici nell'Appendice.

Nella prima fase, si è deciso di reperire i dati per poter classificare i rifiuti utilizzando tre strategie: Web Scraping, programmazione di un Bot di Telegram e costruzione di un cestino che potesse registrare i dati dei rifiuti. Ognuna di queste strategie è stata scelta per ragioni differenti.

Il Web Scraping è stata scelta come strategia iniziale perché il World Wide Web costituisce una fonte quasi infinita di dati (in questo caso di immagini) e non usarlo sarebbe stata una scelta che avrebbe portato all'esclusione di risorse determinanti.

Il Bot di Telegram è stato scelto come strumento per avere immagini più precise riguardo alle macrocategorie più comuni di rifiuti, quali: carta, plastica e vetro.

Infine si è deciso di costruire un cestino così da poter raccogliere dei dati selezionati reputati importanti nel classificare gli oggetti. Come tecnologia di implementazione in questa prima fase si è deciso di utilizzare la piattaforma hardware Arduino per la sua semplicità e versatilità, per la realizzazione di questa parte del progetto è stato necessario costruire un software che fosse in grado di registrare e salvare i dati inviati dall'Arduino così da automatizzare e sveltire il più possibile il processo di costruzione del dataset.

La fase successiva costituisce il fulcro di questo progetto in quanto si sono dovuti analizzare i dati raccolti precedentemente e capire come, sfruttando algoritmi di Deep Learning e Machine Learning, riuscire a classificare nel modo migliore gli oggetti.

Nell'ultima fase si sono combinati tramite meccanismi di Ensemble Learning i vari classificatori così da trovare un previsore migliore.

## 1.2 Come differenziare

Il regolamento della raccolta differenziata può differire da zona a zona. Come regole di riferimento si è deciso di utilizzare il regolamento vigente nella città metropolitana di Milano. Il capoluogo lombardo, infatti, è diventato il Comune che ha raggiunto la quota record del 54% di raccolta differenziata, situandosi in cima alla classifica italiana e al secondo posto in Europa dopo Vienna. Un risultato importante e un fenomeno che ha catturato anche l'attenzione di New York, il cui Assessorato all'ambiente di New York ha deciso di studiare il modus operandi di Milano per adattarlo alla metropoli americana.

Nei prossimi capitoli si parlerà genericamente di plastica, carta o altro rifiuto. In questa sezione descriviamo quali materiali o oggetti rientrano nelle rispettive categorie.

Innanzitutto va ricordato come esistano cinque differenti tipi di rifiuti, da depositare in appositi cassonetti: si tratta del sacco trasparente neutro per i rifiuti generici, il sacco giallo per plastica e metallo, il cassonetto verde per il vetro, il cassonetto bianco per carta e cartone e il cassonetto marrone per rifiuti organici. Vediamo le varie corrispondenze.

- Indifferenziato (sacco trasparente): vi finiscono tutti i rifiuti generici come i piatti rotti, la ceramica in generale, la carta sporca e oleata, cd, DVD, videocassette e musicassette, piccoli accendini, filtri dell’aspirapolvere, pannolini e assorbenti e mozziconi di sigaretta.
- Carta (cassonetto bianco): vi finiscono tutti gli oggetti di carta o cartone quali giornali, riviste, libri e quaderni privati delle parti di plastica, adesive e in metallo, ma anche i contenitori in Tetra Pak (quelli usati per il latte, i succhi di frutta e così via), le scatole in cartone e i cartoni della pizza senza avanzi.
- Plastica (sacco giallo): vi finiscono le plastiche e i metalli, inclusivi di bottiglie, flaconi e sacchetti di plastica, le vaschette per gli alimenti anche in polistirolo, tutte le scatolette e i barattoli per alimenti una volta sciacquate, lattine per bevande, tubetti di plastica (del dentifricio, per esempio), fogli di alluminio, pellicole per imballaggio, nonché oggetti in metallo come pentole, posate, caffettiere, tappi, capsule, chiavi e lucchetti.
- Vetro (cassonetto verde): vi finiscono tutti i materiali in vetro come bottiglie e bicchieri, vasi, caraffe e barattoli. Da non inserirvi invece lampadine, specchi e gli oggetti in cristallo.
- Umido (cassonetto marrone): vi finiscono tutti i rifiuti di natura organica in tutte le loro parti, inclusivi di frutta, frutta secca, verdura, carne e pesce, pane, riso, pasta, scarti di cucina, avanzi, fondi di caffè, filtri di tè e tisane, fiori, semi e foglie, e alimenti avariati.

## 1.3 Web Scraping

### 1.3.1 Funzionamento

Il Web scraping [3] (chiamato anche web harvesting, o web data extraction) è un insieme di tecniche utilizzate per l’estrazione di dati dai siti Web. È una forma di copia, in cui i dati specifici vengono raccolti e copiati dal web accedendo al World Wide Web direttamente utilizzando il protocollo Hypertext

Transfer Protocol o tramite un browser Web, in genere in un database locale centrale o foglio di calcolo, per il successivo recupero o analisi, il termine in genere si riferisce a processi automatizzati implementati utilizzando un web crawler, un bot o un server.

Tra le tante possibili soluzioni si è deciso di utilizzare la tecnica di Web Scraping chiamata HTML Parsing ossia una deserializzazione delle pagine HTML. Questo processo riceve il codice HTML non elaborato, lo interpreta e genera dal codice una struttura ad albero DOM (Document Object Model).

Questa tecnica è molto versatile in quanto gran parte dei siti Web hanno grandi raccolte di pagine generate dinamicamente da un'origine strutturata sottostante come un database. I dati della stessa categoria sono in genere codificati in pagine simili da uno script o un modello comune. Nel data mining, un programma che rileva tali modelli in una particolare fonte di informazioni, ne estrae il contenuto e lo traduce in un modulo relazionale, viene chiamato wrapper. Gli algoritmi di generazione dei wrapper presuppongono che le pagine di input di un sistema di induzione di wrapper siano conformi a un modello comune e che possano essere facilmente identificate in termini di schema comune URL.

### 1.3.2 Implementazione

Per fare Web Scraping si è deciso di utilizzare il linguaggio R, questo perchè oltre ai pacchetti appositi che aiutano nell'operazione di creazione dello wrapper in R sono presenti delle ottime librerie per fare text mining, la più famosa ““stringr””, che aiutano in determinate operazioni a gestire operazioni di HTML parsing più complicate.

La libreria utilizzata in questa fase è ”rvest” (creata da Hadley Wickham [aut, cre], RStudio [cph]) la quale gestisce e migliora alcune funzioni contenute nel pacchetto ”xml2” (creata da Hadley Wickham [aut], Jim Hester [aut, cre], Jeroen Ooms [aut], RStudio [cph], R Foundation [ctb]), queste due librerie si basano su un pacchetto creato per il linguaggio C++ da Daniel Veillard (versione definitiva pubblicata alla fine del 2012) di nome ”libxml2”. Le funzioni che abbiamo utilizzato di queste librerie sono tre: *read\_html()* del pacchetto ”xml2”; *xml\_attrs* e *html\_nodes()* del pacchetto ”rvest”.

La prima funzione `read_html()` serve per deserializzare le pagine HTML e trasformarle in una lista strutturata come un albero DOM.

---

```
install.packages("xml2")
library("xml2")
page<-read_html("###---link---###")
```

---

Le due funzioni del pacchetto "rvest", invece, aiutano ad individuare i diversi elementi del documento. In questo caso si utilizzano congiuntamente per trovare il link di origine delle immagini per poi scaricarle.

---

```
install.packages("rvest")
library("rvest")
t<-1 # img t-esima presente nella pagina
link_img<-xml_attrs(html_nodes(page, xpath =
  "//img")[[t]][["src"]])
```

---

Infine una volta trovati i link delle immagini presenti nelle varie pagine si salvano tramite la funzione `download.file()`

---

```
download.file(link_img, destfile="PATH", method='curl')
```

---

Per la ricerca dei siti dove scaricare le immagini si è utilizzata la logica alla base delle ricerche sui più comuni motori di ricerca per immagini: Google, Bing e Pixabay.

La logica che sta alla base delle ricerche online permette di trovare tutte le immagini collegate inserendo una o più parole chiave. Quindi si è deciso di creare delle liste di parole per ogni categorie specificate nella sezione precedente e di trovare le immagini collegate al quella particolare Keyword.

In termini generali si è creato un array con una serie di parole chiave queste sono state combinate all'interno di un ciclo `for` creando, attraverso le funzioni `gsub()` e `paste0()` creando un URL che potesse comunicare con il motore di ricerca.

Le parole chiavi scelte per l'analisi:

---

```
Plastica<-c("Plastica", "bottiglie di plastica", "sacchetti di
plastica", "tappi di plastica", "giochi di plastica", "utensili
di plastica", "pacchi di plastica", "posate di plastica",
```

"bicchieri di plastica", "vaschette di plastica", "tubetti dentifricio", "pellicola cucina", "sedie di plastica", "secchi di plastica", "vaschette plastica", "contenitore uova di plastica")

Carta<-c("Carta", "carta giornali", "fogli di carta", "contenitore uova di carta", "tovaglioli di carta", "pacchi di cartone", "cartone pizza", "posate di carta", "bicchieri di carta", "libri di carta", "quaderni di carta", "sacchetti di carta", "tovaglie di carta", "scatole di cartone", "riviste di carta", "carta appallottolata")

Tetra Pak<-c("Tetra Pak latte", "Tetra Pak succhi", "contenitori in tetra pak", "tetra pak succo brico ")

Polistirolo<-c("contenitori in polistirolo", "polistirolo", "vaschette polistirolo")

Vetro<-c("Vetro", "Bottiglie di vetro", "tazzine di vetro", "vetro rotto", "vetro bottiglie vino", "vetro bottiglie birra", "vetro bottiglie alcolici", "vetro bottiglie bevante", "caraffe di vetro", "calici di vetro", "vasi di vetro", "bicchieri di vetro", "bicchieri di vetro rotti", "lastre di vetro", "contenitori di vetro")

Indifferenziato<-c("ceramica", "carta sporca e oleata", "cd", "dvd", "videocassette", "musicasette", "accendini", "pannolini", "assorbenti", "mozziconi di sigarette", "lampadine", "cristallo", "specchi", "vasi ceramica", "filtri dell'aspirapolvere")

Legno<-c("legno", "legname", "giochi di legno", "utensili di legno", "assi di legno", "posate di legno", "ciotole di legno", "mestoli di legno", "sedie di legno", "tavoli di legno", "piatti di legno", "scaffali di legno", "mobili di legno", "tagliere di legno", "tappi di sugero")

Metalli<-c("ferro", "metallo", "pentole", "mestoli di ferro", "lastre metallo", "chiavi di ferro", "utensili di ferro", "ferro battuto", "padelle", "posate di metallo", "forchette di metallo", "coltelli di metallo", "cucchiai di metallo", "caffettiere", "Lucchetti di ferro")

Umido<-c("scarti cibo", "umido cibo", "organico cibo", "scarti pesce", "scarti carne", "fondi di caffè", "frutta secca", "avanzi cibo", "filtrati te e tisane", "pane", "fiori e foglie", "alimenti avariati", "scarti di cucina", "avanzi pasta", "avanzi di riso")

```
Latta_e_alluminio<-c("lattine per bevande", "fogli di alluminio",
  "lattine di cocacola", "lattine di pepsi", "lattine redbull",
  "lattine di sprite", "lattine di 7up", "lattine di fanta",
  "lattine di birra", "lattine di pelati", "lattine di cereali",
  "lattine di legumi", "alluminio per cucina", "teglie di
  alluminio", "lattine")
```

---

La funzione per scaricare le immagini:

---

```
#1
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(" ", "+",parola_chiave[i]), "+jpg&source=lnms&tbo=isch")))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
  txt<-c(txt,xml_attrs(html_nodes(page, xpath =
    "//img"))[[t]])[["src"]]
})
# 2
try({page<-read_html(paste0("https://www.bing.com/images/search?q="
  ,gsub(" ", "+",parola_chiave[i]), "+jpg&FORM=HDRSC2")))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
  txt<-c(txt,xml_attrs(html_nodes(page, xpath =
    "//img"))[[t]])[["src"]]
})
# 3
try({page<-read_html(paste0("https://pixabay.com/it/images/search/"
  ,gsub(" ", "%20",parola_chiave[i])))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
  txt<-c(txt,xml_attrs(html_nodes(page, xpath =
    "//img"))[[t]])[["src"]]
}})
```

---

Così facendo si è stati in grado di trovare i links di tutte le immagini presenti nel sito, nella funzione sopra descritta si sono spesso utilizzati dei `try()` per evitare gli errori 404, molto comuni quando si automatizza il parsing HTML. Funzione al completo descritta nell'Appendice A.1.1 .

## 1.4 Bot Telegram

### 1.4.1 Funzionamento

Telegram è una applicazione di messaggistica gratuita creata e ideata a Berlino da Pavel Durov e Nikolai Durov. La prima particolarità di Telegram è quella di basarsi su un protocollo di comunicazione completamente open source, sviluppato per ridurre al minimo la quantità di byte inviati per ogni messaggio. Questa soluzione permette di avere una maggiore velocità anche in condizioni di scarsa recezione. La seconda qualità di Telegram riguarda l'elevato livello di sicurezza, infatti, grazie all'utilizzo di algoritmi di criptazione interni alle chat aumenta notevolmente il livello di privacy.

Secondo la definizione classica data al termine Bot (abbreviazione di "robot"), è una chat con la quale è possibile inviare input ad un server tramite comandi precedentemente programmati e ricevere output direttamente nella chat e/o sul server. I chatbot stanno diventando sempre più importanti in un mondo nel quale si cerca di automatizzare il più possibile ogni tipo di processo e anche le grandi aziende si stanno spingendo verso quella direzione si pensi ad esempio a "TOBI" chatbot di Vodafone con il quale è possibile chattare sull'app o sul sito della società di telecomunicazioni per aiutare il cliente a risolvere eventuali problemi.

### 1.4.2 Implementazione

Il Bot implementato in questa sezione ha come funzione quella di ricevere immagini per categoria di rifiuto, salvarle e restituire a colui che ha inviato l'immagine un messaggio di ringraziamento. Nel programmare tale Bot si è utilizzato il pacchetto di Python "Telepot" grazie al quale si riesce ad inviare e ricevere (GET e POST request nel linguaggio PHP) richieste al server direttamente dalla chat. In particolare questa libreria permette di identificare la tipologia di messaggio che il soggetto invia sulla chat tramite la funzione "content\_type" e istruirlo a compiere determinate azioni nel caso vengano inviate determinate tipologie di messaggio. Il bot in questione doveva salvare l'immagine con la funzione "bot.download\_file(file, path)" e restituire un mes-

saggio di ringraziamento attraverso la funzione ”bot.sendMessage(chat\_id, message)”.

---

```
if content_type == 'photo':  
    bot.download_file(msg['photo'][1]['file_id'], 'dati/image'+  
        str(datetime.now())[0:19] +'.png')  
    bot.sendMessage(chat_id, 'Thanks for send me the photo! See you  
soon!')
```

---

Si è deciso di creare 3 bot per le tre categorie principali di rifiuti: Plastica, Carta e Vetro.

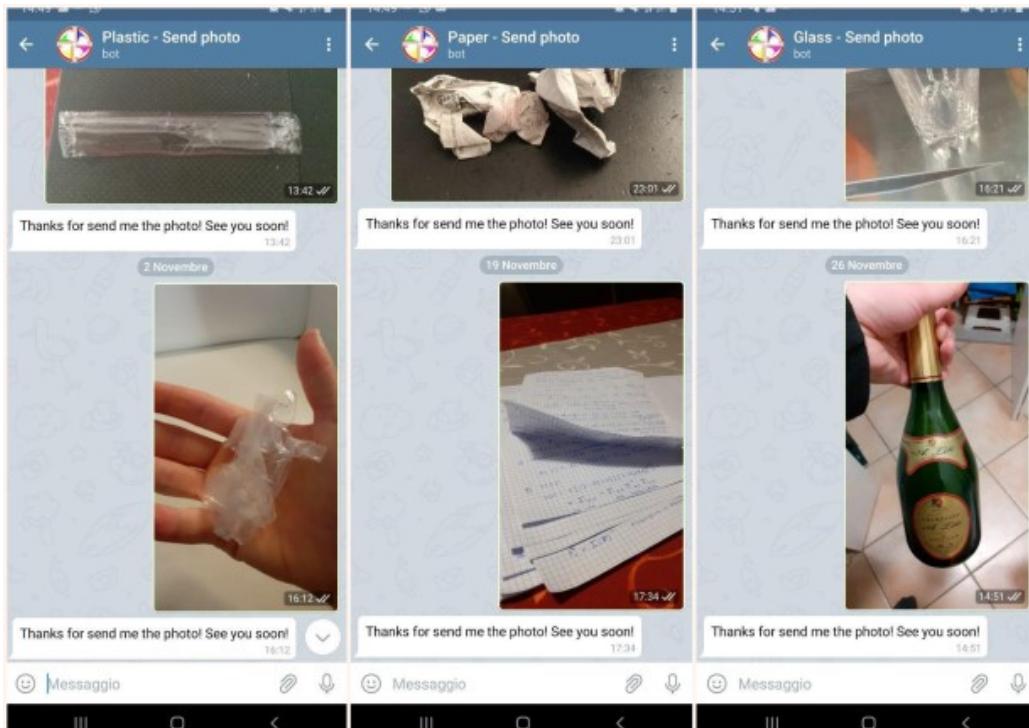


Figura 1.1: Funzionamento Bot di Telegram

Per il funzionamento del Bot si è deciso di utilizzare un server collegato ad un rete locale piuttosto che un VPS (Virtual Private Server) per due ragioni: da un lato, non avendo la necessità di una grande capacità computazionale,

risultava più conveniente in termini economici e dall'altro erano più semplici eventuali modifiche in corso d'opera.

## 1.5 Il cestino

Come ultima fonte per reperire i dati si è deciso di costruire un cestino così da poter raccogliere dati di tipo strutturato sui rifiuti. Nel costruire il cestino si è dovuto progettare un lavoro di tipo meccanico per far sì che il cestino funzionasse dal punto di vista strutturale; di tipo robotico in quanto questo cestino doveva essere in grado di muoversi e inviare i dati ricevuti dai sensori al computer; e, infine, di tipo softwaristico in quanto si è dovuto programmare un software che fosse in grado di in fase iniziale ricevere e salvare i dati inviati dal cestino e in una fase successiva riconoscere l'oggetto attraverso i dati ricevuti e riuscire a comandare il cestino per fargli fare la differenziata.

### 1.5.1 Meccanica

Nel costruire il cestino per semplicità si è pensato di dividere la struttura in due unità: la struttura inferiore (o struttura reggente) che ha come scopo quello di contenere i comparti delle categorie di rifiuto e di reggere l'assetto; la struttura superiore (o il "cervello") ossia una telaio mobile dove vengono posti tutti i componenti elettronici che hanno come scopo quello di interpretare i messaggi inviati dal cestino, interfacciarsi con l'utente e fare la raccolta differenziata ruotando sopra la struttura reggente.

La struttura inferiore è composta da una base circolare di dimensioni 40x40x5 cm da cui si ergono 4 pilastri di dimensione 1x1x20 cm che sorreggono un cerchio di raggio 40 cm e di dimensione della linea 5 cm su cui vengono poste delle ruote che permettono alla struttura superiore di muoversi in fluidità.

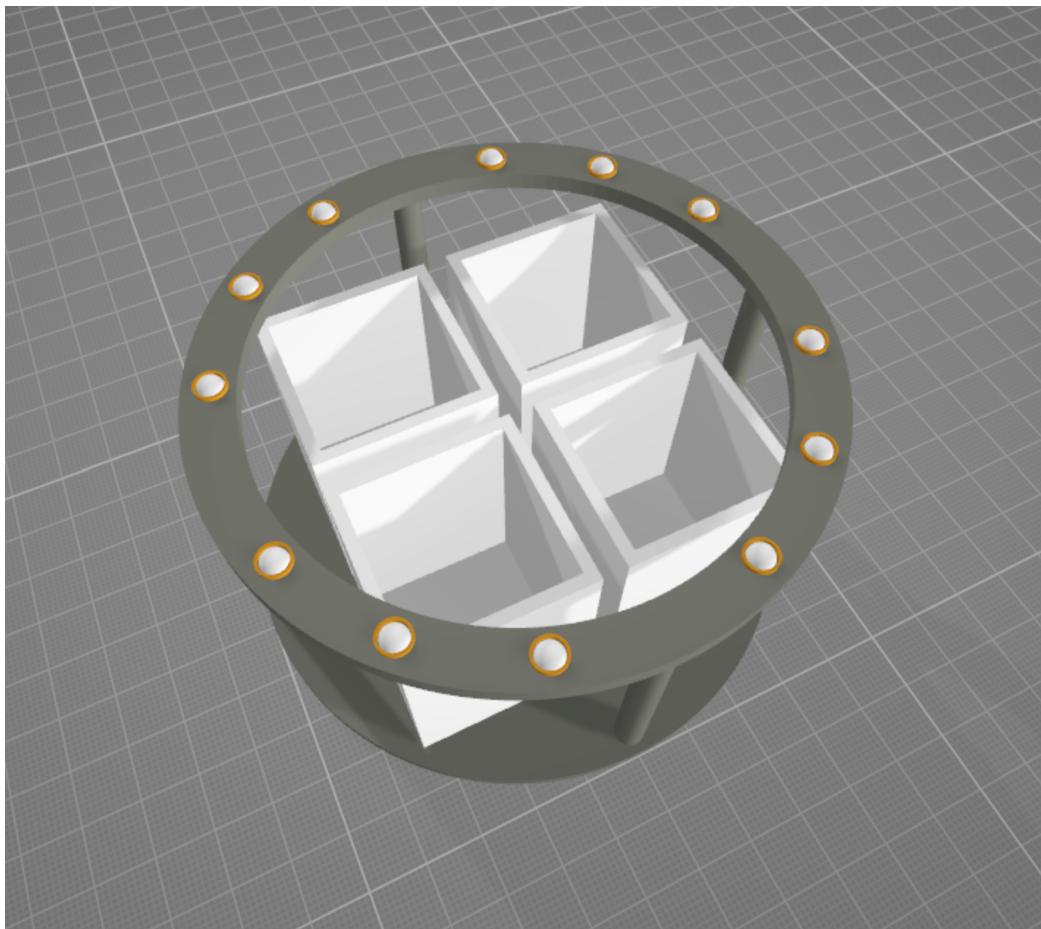


Figura 1.2: Modello 3D della struttura inferiore

In fase di costruzione si è deciso di diminuire il numero di ruote scorrevoli poste sopra in quanto si reputavano superflue, al posto di 12 si è preferito collocarne 8. La base della struttura è stata costruita in legno , il resto della struttura in ferro.

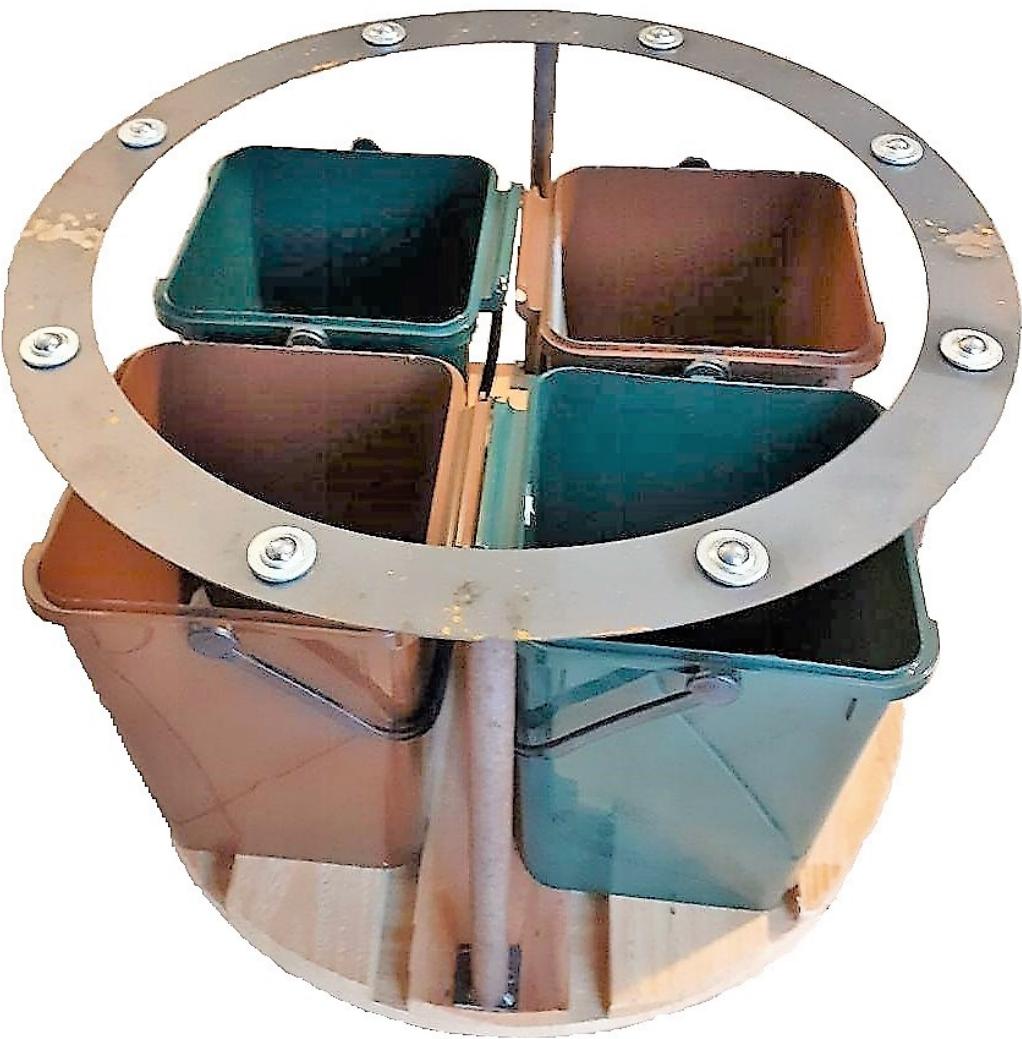


Figura 1.3: Foto della struttura inferiore

La struttura superiore costituisce la base sopra la quale vengono poste tutte le componenti di robotica, essa è costituita da una base circolare 40x40x1 cm sopra alla quale è agganciato un cestino 10x15x25 cm con all'interno una base mobile delle dimensioni interne del cestino che permette di far cadere i rifiuti. Sotto la base sono posizionate su un triangolo equilatero inscritto nella circonferenza posta sugli estremi della base tre ruote, due di queste

girano a vuoto per centrare la base sulla struttura inferiore mentre una è robotizzata e consente all’impianto di ruotare.

La base sopra è divisa in tre sezioni: la parte con le componenti di robotica che inviano e ricevono segnali, la parte dove è posto un mini pc Beelink AP34 (con processore Intel Celeron N3450 e 4 GB di memoria RAM DDR3 SDRAM) sul quale viene installato il software in grado di ricevere e inviare input e output al sistema e infine una terza sezione dove è posizionato un monitor 3,5 pollici touch screen che permette all’utente di interfacciarsi con il sistema attraverso il programma.

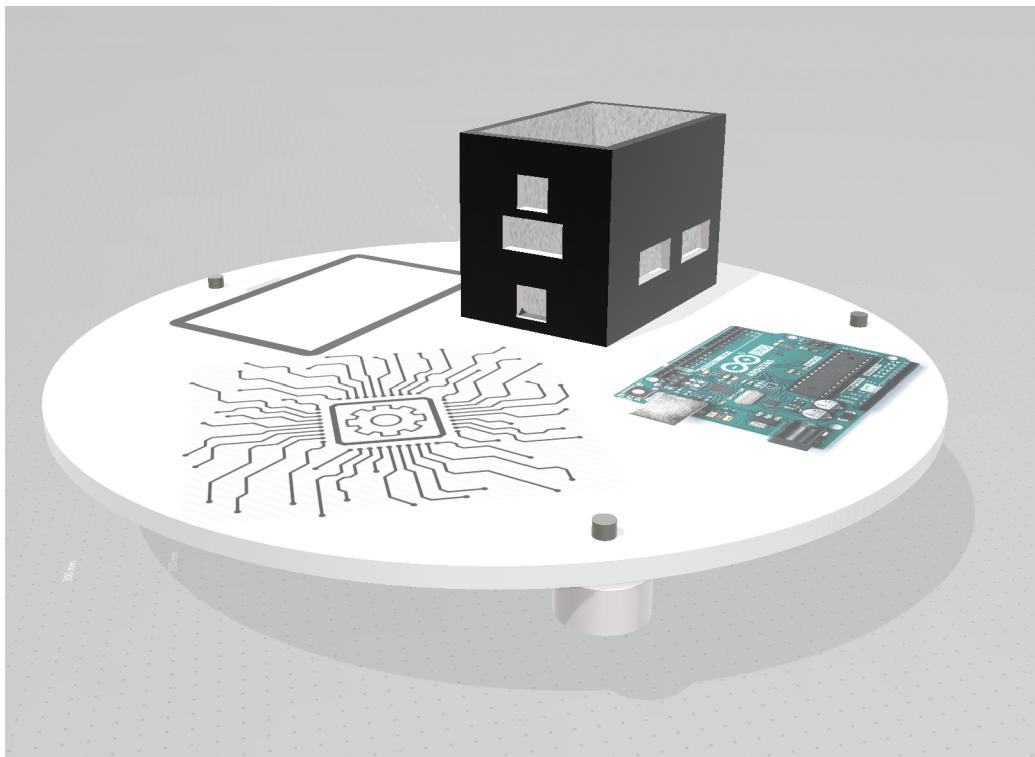


Figura 1.4: Modello 3D della struttura superiore

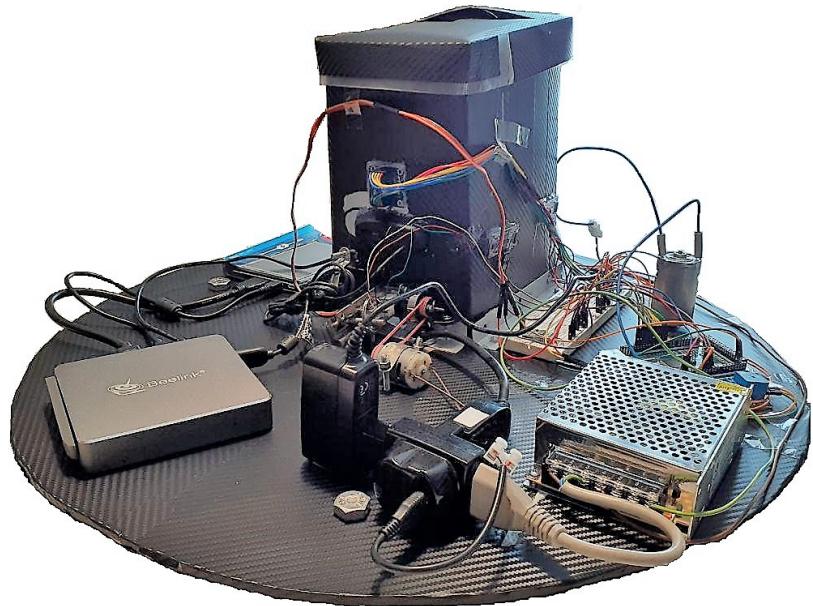


Figura 1.5: Foto della struttura superiore

### 1.5.2 Robotica

Arduino unisce due mondi: quello hardware [4] [5], rappresentato dalla scheda e dai componenti ad essa collegabili e quello software, rappresentato dal programma scritto e caricato all'interno della componente fisica. La scheda che si è deciso di utilizzare è la Arduino Mega 2560 essa è composta da un microcontrollore ATmega2560 [6], di 54 pin digitali e 16 analogici, 4 porte seriali UART, un cristallo oscillatore a 16 MHz, una porta USB e un jack di alimentazione, un header ICSP e un pulsante di reset. Ha tre tipi di memoria: Flash, SRAM ed EPROM. La scheda lavora ad una tensione nominale di 5V e sopporta una corrente massima di 40 mA.

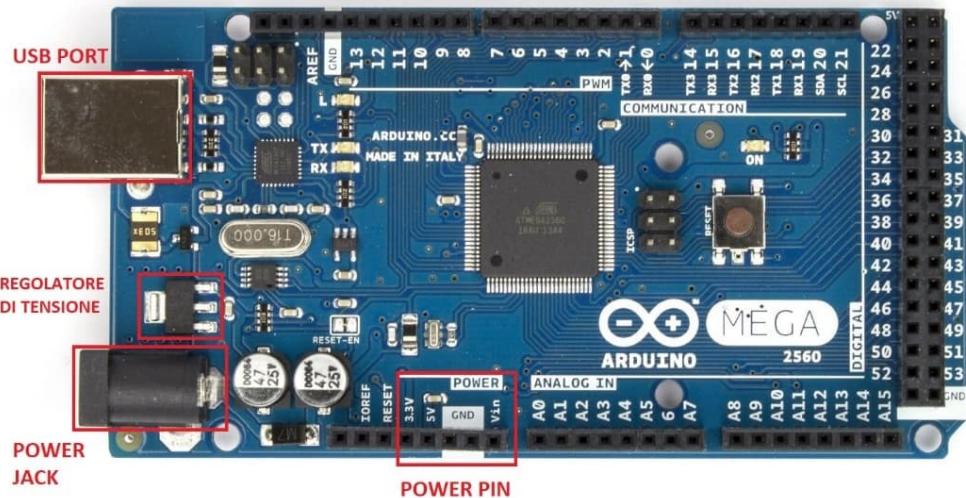


Figura 1.6: Scheda Arduino Mega 2560

I componenti collegabili ad Arduino possono essere classificati in quattro macrocategorie [7]:

- I sensori

I sensori sono componenti elettronici in grado di percepire e misurare le caratteristiche fisiche dell’ambiente circostante e quindi, ad esempio, luminosità, temperatura, umidità, suono, movimento, campo magnetico ed elettromagnetico.

- Gli attuatori

Gli attuatori sono componenti in grado di modificare le caratteristiche fisiche dell’ambiente circostante e quindi essenzialmente, sorgenti di luce, calore, umidità, suono, movimento, campo magnetico ed elettromagnetico.

- I componenti complessi

Sono dei circuiti, dotati di microprocessore e componenti, in grado di fornire un servizio e quindi da fungere contemporaneamente, da sensori ed attuatori, come, ad esempio, i sistemi di gestione delle connessioni bluetooth, che possono ricevere ed inviare informazioni ad arduino, oppure la stazione di lettura/scrittura di secure digital.

- I componenti di supporto

Sono componenti che supportano l'operatività di sensori ed attuatori. Tra i componenti di supporto più comuni troviamo le resistenze, i condensatori, i fusibili ed altri ancora come la breadboard, gli shield, i pulsanti ed i cavi di collegamento

In questa sezione non si andranno a descrivere tutti gli elementi inseriti all'interno del cestino ma si andranno ad approfondire i quattro componenti usati per ricavare le informazioni dai rifiuti inseriti nel cestino.

**Una fotoresistenza** è una resistenza la cui impedenza, ossia la cui capacità di far circolare elettricità, varia al variare della luce che la colpisce. All'aumentare della luce diminuisce la resistenza, e viceversa. Tipicamente è un sensore di tipo analogico. Per utilizzarlo si collega una gamba ad una porta analogica e, in parallelo, ad una resistenza da 10k ohm collegata a terra mentre si collega l'altra gamba all'alimentazione da 5 volt. La porta analogica restituisce un valore da 0 a 1023 che varia al variare della luce che colpisce la fotoresistenza. Più la luce è forte, più il valore si avvicina a 1023



Figura 1.7: Fotoresistenza

---

```
void setup() {
    // put your setup code here, to run once:
    pinMode(A2, INPUT);
    pinMode(A3, INPUT);
    Serial.begin(9600);
}
```

```

void loop() {
    // put your main code here, to run repeatedly:
    luce1=analogRead(A2);
    luce2=analogRead(A3);

    Serial.print(luce1);
    Serial.print(",");
    Serial.print(luce2);
    Serial.println(";");
}

```

---

**Il modulo ad ultrasuoni HC-SR04** viene normalmente utilizzato per rilevare eventuali ostacoli e misurarne la distanza (da 2 a 400 cm). Il modulo opera usando la medesima tecnica di rilevamento utilizzata, in natura, dai pipistrelli, è formato da un generatore di ultrasuoni, da un ricevitore e da un circuito di controllo. Il modulo si avvia quando riceve un impulso di almeno 10 microsecondi attraverso il “trig pin”, ossia la porta di attivazione. A questo punto lancia una serie di otto onde sonore da 40 kHz e si mette in attesa di un segnale di ritorno. Appena lo riceve attiva la porta di uscita (echo pin) e la mantiene attiva per un tempo proporzionale al tempo intercorso tra l’invio del segnale acustico ed il suo ritorno Conoscendo la velocità del suono e sapendo che il “viaggio” dell’onda sonora è stato il doppio della distanza tra il modulo e l’ostacolo (l’onda è andata dal generatore all’ostacolo e da qui’è tornata al sensore) la distanza è derivata dalla seguente formula:

$$distanza = tempo \cdot 340/2$$

dove: distanza sono i metri tra il modulo HC-SR04 e l’ostacolo, tempo sono i secondi di attivazione della porta di uscita , 340 è la velocità del suono in metri al secondo.



Figura 1.8: Il modulo ad ultrasuoni HC-SR04

---

```
#include <NewPing.h>

NewPing sonar1(10, 9, 200);
NewPing sonar2(12, 11, 200);

void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);
}

void loop() {
    // put your main code here, to run repeatedly:

    unsigned int distanza1 = sonar1.ping();
    distanza1=distanza1 / US_ROUNDTRIP_CM;

    unsigned int distanza2 = sonar2.ping();
    distanza2=distanza2 / US_ROUNDTRIP_CM;

    Serial.print(distanza1);
    Serial.print(",");
    Serial.print(distanza1);
    Serial.println(";");
}
```

---

**I sensori di peso** sfruttano la variazione di resistenza elettrica che alcuni materiali manifestano quando sono sottoposti a compressione o a trazione. Un sensore di peso ha la forma di una barra di metallo caratterizzata da due grandi fori aventi lo scopo di facilitarne la flessione nel momento in cui, su uno dei due estremi, viene esercitata una forza. Gli elementi che permettono il funzionamento di questa tecnologia sono gli estensimetri che, opportunamente posizionati sulla barra forata forniscono indicazioni sufficienti a dimensionare la sollecitazione. L'estensimetro elettrico a resistenza è costituito da una griglia di sottilissimo filo metallico rigidamente applicata su di un supporto di materiale plastico. L'estensimetro viene utilizzato incollandolo sulla superficie del corpo di cui si vogliono misurare le deformazioni. Il filo segue le deformazioni della superficie a cui è incollato, allungandosi ed accorciandosi insieme ad essa; queste variazioni dimensionali causano una

variazione della resistenza elettrica del filo. Misurando tali variazioni, si può risalire all'entità della deformazione che le ha causate. La variazione di resistenza, interpretata da un apposito driver **la scheda HX711**, consente ad Arduino di formulare precise indicazioni sulla sollecitazione cui la barra è sottoposta. Esistono diversi sensori di peso che hanno portate sensibilmente differenti, in questo progetto è stato utilizzato un sensore la cui portata massima è 20kg. Con questo componente utilizzeremo una libreria che restituisce il peso in Ounce quindi per trasformarlo in grammi occorre moltiplicare il valore restituito per il cambio ossia 28.3495231.

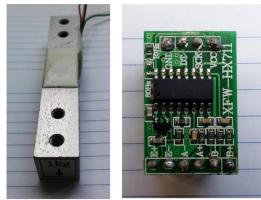


Figura 1.9: A sinistra sensore del peso a destra scheda HX711

---

```
#include "HX711.h"
// HX711.DOUT - pin #A4
// HX711.PD_SCK - pin #A5
HX711 scale(A4, A5);

void setup() {
    // put your setup code here, to run once:
    // set HX711
    scale.set_scale(2280.f);
    scale.tare();
    Serial.begin(9600);
}

void loop() {
    // put your main code here, to run repeatedly:
    Serial.print(scale.get_units(10)*28.3495231, 1);
}
```

---

**Il sensore TCS3200** è un componente utilizzato allo scopo di individuare quali colori sono presenti nel suo campo visivo e esprimere secondo percentuali. Si tratta di un sensore composto da una matrice di 64 elementi fotosensibili di cui 16 con un filtro rosso 16 con un filtro verde e 16 con un filtro blu, più altri 16 non filtrati. Questo sensore all'uscita produce una frequenza che è funzione della luce che lo colpisce. E' possibile selezionare l'uscita legandola alla misura di uno dei 4 gruppi di sensori descritti. Arduino seleziona il colore da misurare e legge la frequenza in uscita dal modulo TCS230. Si è usato una funzione per leggere i colori chiamata "readColor()".



Figura 1.10: ColorSensor TCS3200

---

```
//Read-Color Function
int readColor() {

    //Setting red filtered photodiodes to be read
    digitalWrite(s2, LOW);
    digitalWrite(s3, LOW);

    //Reading the output frequency
    frequency = pulseIn(out, LOW);
    int R = frequency;

    //Printing the value on the serial monitor
    Serial.print(frequency); //printing rosso color frequency
    Serial.print(",");
    Serial.print(" ");

    //Setting Green filtered photodiodes to be read
    digitalWrite(s2, HIGH);
    digitalWrite(s3, HIGH);
```

```

//Reading the output frequency
frequency = pulseIn(out, LOW);
int G = frequency;

//Printing the value on the serial monitor
Serial.print(frequency);
//printing verde color frequency
Serial.print(",");
Serial.println();

//Setting Blue filtered photodiodes to be read
digitalWrite(s2, LOW);
digitalWrite(s3, HIGH);

//Reading the output frequency
frequency = pulseIn(out, LOW);
int B = frequency;

//Printing the value on the serial monitor
Serial.print(frequency); //printing blu color frequency
Serial.print(",");
Serial.println();

if(R<260 & R>230 & G<860 & G>800){
    color = 1; // Rosso
}
if(G<420 & G>370 & B<350 & B>305){
    color = 2; // Blu
}
if(R<450 & R>420 & G<420 & G>390){
    color = 3; // Verde
}
return color;
}

```

---

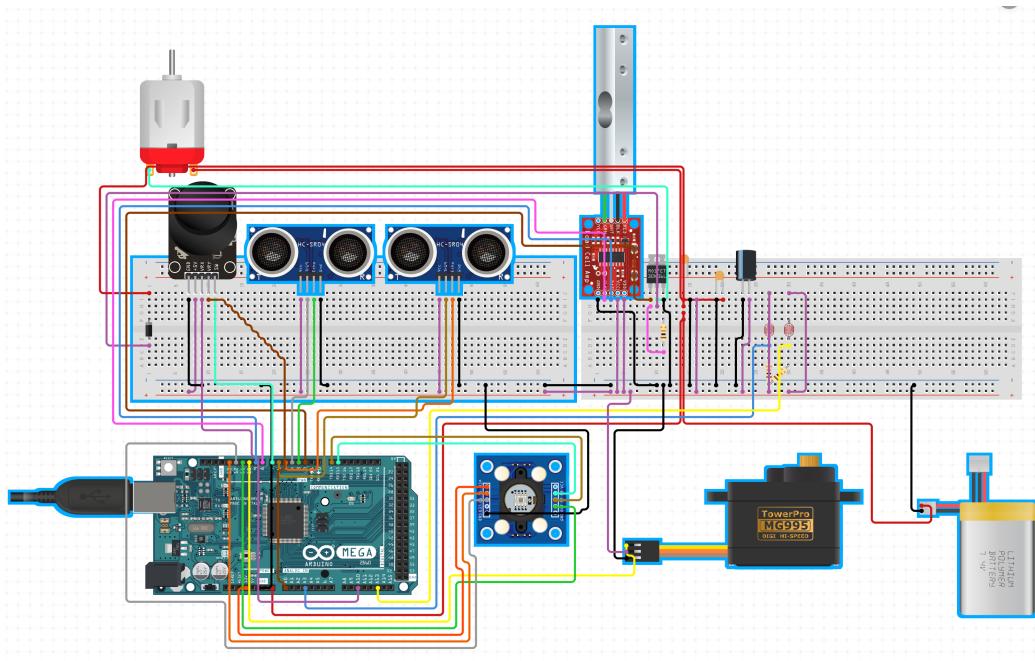


Figura 1.11: Lo schema di collegamento del cestino (in aggiunta a questo sistema c'è un motore DC 12 V controllato con un pin digitale e collegato ad un Power Supply 12V 10mA che serve a far ruotare la struttura superiore)

### 1.5.3 Softwaristica

In questo capitolo non si andranno ad analizzare i codici nello specifico in quanto molto lunghi e poco utili ai fini della tesi lasciando i codici commentati nell'Appendice A sezione [A.1.4] e [A.3.1] ma si spiegheranno le funzioni che il software deve svolgere.

Il programma lo ho chiamato "Recycling bin" ed è strutturato in 3 pagine. La prima pagina denominata "Form1" serve per indirizzare l'utente verso la funzione di addestramento e reperimento dei dati o la sezione di riconoscimento e analisi dei dati.

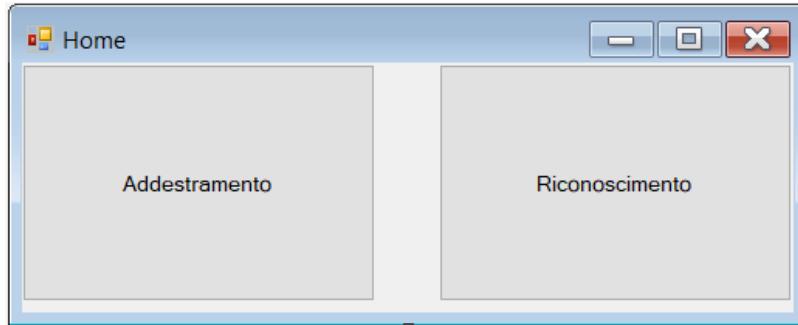


Figura 1.12: Form1

La seconda pagina chiamata "Form2" serve per registrare i dati relativi ai rifiuti. Per inizializzare il programma occorre prima collegarlo alla telecamera del cestino una Logitech C270 Webcam HD scegliendo le specifiche grafiche e poi selezionare la porta COM relativa all'Arduino, operando antecedentemente una scansione delle porte così da poter leggere ed inviare i dati.

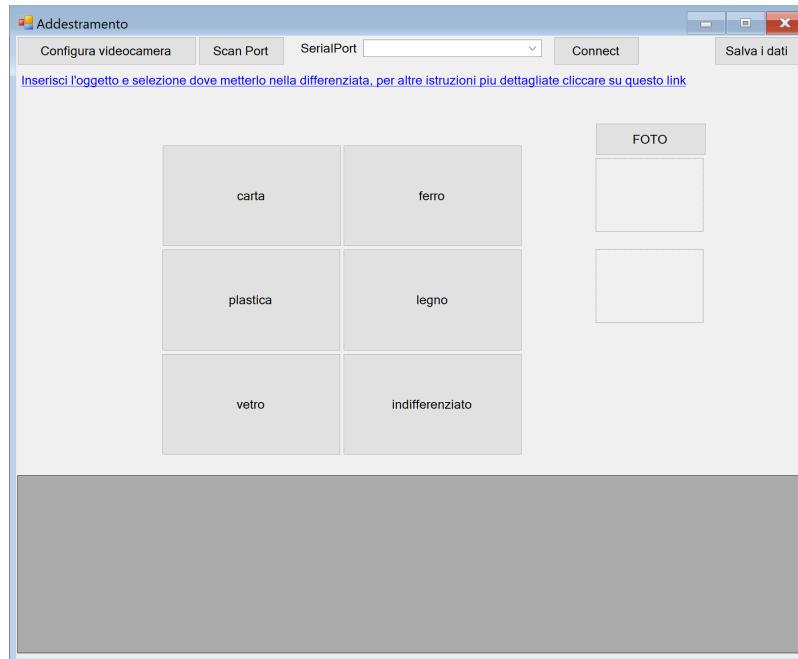


Figura 1.13: Form2

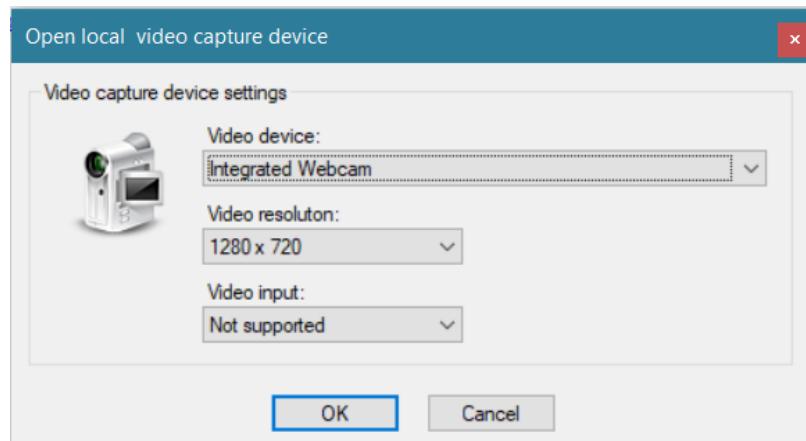


Figura 1.14: Form di inizializzazione Webcam

Nella terza pagina indicizzata come "Form3" bisogna inizializzarla come visto per la precedente e poi una volta inserito l'oggetto cliccare sul pulsante riconosci per attivare il cestino al riconoscimento.



Figura 1.15: Form3

# **Capitolo 2**

## **Addestramento train e modelli**

### **2.1 Descrizione tipologie di dati**

descrivere tutti i dati che il computer registra o riceve

Dico come introduzione quanti dati ho raccolto e metto un po' di statistiche descrittive di base

(Dati non strutturati: CNN, Dati strutturati: ML [testare un po']) spiegazione teorica e applicazione

### **2.2 Dati Strutturati**

#### **2.2.1 Spiegazione teorica dei modelli utilizzati**

#### **2.2.2 Applicazione dei modelli**

### **2.3 Dati non strutturati**

#### **2.3.1 Spiegazione teorica dei modelli utilizzati**

#### **2.3.2 Applicazione dei modelli**

# **Capitolo 3**

## **Verifica test, confronto dei modelli**

Verifica test, confronto dei modelli (specificità vs accuratezza) e Ensemble Learning

### **3.1 Confronto modelli**

### **3.2 Ensemble Learning**

#### **3.2.1 Spiegazione teorica dei modelli utilizzati**

#### **3.2.2 Applicazione dei modelli**

Utilizzerei modelli Staked:  
Staked regression algorithm

### **3.3 Esempi pratici di applicazioni**

#### **3.3.1 Bot di Telegram**

#### **3.3.2 Cestino**

# Conclusioni

- pregi del lavoro
  - punti critici
  - possibili miglioramenti (qui soffermarsi un po')
  - Amazon Web Services
  - Lettura bar code Scrivere conclusioni sull'efficacia del modello

# Appendice A

## Codici

### A.1 Reperimento dei dati

#### A.1.1 Codice Web Scraping (R)

---

```
library(xml2)
library(rvest)
library(stringr, warn.conflicts = F)

webscraing_img <-function(img){
  txt<-"img"
  for(i in 1:length(img)){
    # 1
    try({page<-read_html(paste0("https://www.google.com/search?source=&q="
      ,gsub(" ", "+",img[i]), "+jpg&source=lnms&tbo=isch")))
    for (t in 1:length(html_nodes(page,xpath = '//img'))){
      txt<-c(txt,xml_attrs(html_nodes(page, xpath =
        "//img"))[[t]])[["src"]]
    })
    # 2
    try({page<-read_html(paste0("https://www.bing.com/images/search?q="
      ,gsub(" ", "+",img[i]), "+jpg&FORM=HDRSC2")))
    for (t in 1:length(html_nodes(page,xpath = '//img'))){
      txt<-c(txt,xml_attrs(html_nodes(page, xpath =
        "//img"))[[t]])[["src"]]
    })
  }
}
```

```

# 3
try({page<-read_html(paste0("https://pixabay.com/it/images/search/"
  ,gsub(" ", "%20", img[i])))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1a
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
  ", "+,img[i]), "+jpg&source=lnms&tbo=isch&tbs=isz:l"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1b
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
  ", "+,img[i]), "+jpg&source=lnms&tbo=isch&tbs=isz:m"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1c
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
  ", "+,img[i]), "+jpg&source=lnms&tbo=isch&tbs=isz:s"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.1
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
  ", "+,img[i]), "+jpg&tbas=0&tbo=isch&tbs=ic:specific,isc:white"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.2

```

```

try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:teal"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.3
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:black"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.4
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:green"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.5
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:red"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.6
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:blue"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.7

```

```

try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:gray"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.8
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:yellow"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.9
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:pink"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
# 1.10
try({page<-read_html(paste0("https://www.google.com/search?source=&q="
  ,gsub(
    ", "+,img[i]), "+jpg&tbas=0&tbm=isch&tbs=ic:specific,isc:brown"))
for (t in 1:length(html_nodes(page,xpath = '//img')))) {
txt<-c(txt,xml_attrs(html_nodes(page, xpath =
  "//img")[[t]])[["src"]])
}})
}
d<-substr(txt,1,3)=="htt"
immaginidascaricare<-txt[d]
return(immaginidascaricare)
}

```

```




---



```

### A.1.2 Codice Bot Telegram per salvare i dati (Python)

```

import telepot
from datetime import datetime
import time


def rispondi(msg):
    content_type, chat_type, chat_id = telepot.glance(msg)
    if content_type == 'photo':
        bot.download_file(msg['photo'][1]['file_id'], 'dati/image'+
            str(datetime.now())[0:19] +'.png')
        bot.sendMessage(chat_id, 'Thanks for send me the photo! See
            you soon!')

bot =telepot.Bot('####---TOKEN---###')
bot.message_loop(rispondi)

print 'I am listening ...'

```

```
while 1:  
    time.sleep(10)
```

---

### A.1.3 Codice Arduino (C++)

```
#include <NewPing.h>  
#include <Servo.h>  
#include "HX711.h"  
  
// HX711.DOUT - pin #A4  
// HX711.PD_SCK - pin #A5  
  
HX711 scale(A4, A5); // parameter "gain" is omitted; the default  
// value 128 is used by the library  
  
  
NewPing sonar1(10, 9, 200);  
NewPing sonar2(12, 11, 200);  
int magnetic, md, luce1, luce2, distanza1 ;  
const int s0 = 3;  
const int s1 = 4;  
const int s2 =7;  
const int s3 = 5;  
const int out = 6;  
const int motorpin = 52;  
Servo motoreservo;  
int frequency = 0;  
int color=0;  
  
void setup() {  
    // put your setup code here, to run once:  
  
    pinMode(A2,INPUT);  
    pinMode(A3,INPUT);  
    pinMode(A0,INPUT);  
  
    Serial.begin(9600);
```

```

pinMode(s0, OUTPUT);
pinMode(s1, OUTPUT);
pinMode(s2, OUTPUT);
pinMode(s3, OUTPUT);
pinMode(out, INPUT);
motoreservo.attach(motorpin);
digitalWrite(s0, HIGH);
digitalWrite(s1, LOW);

// set HX711
scale.set_scale(2280.f);
scale.tare();
}

void loop() {
    // put your main code here, to run repeatedly:

    magnetic=analogRead(A0);
    luce1=analogRead(A2);
    luce2=analogRead(A3);
    unsigned int distanza1 = sonar1.ping();
    distanza1=distanza1 / US_ROUNDTRIP_CM;

    unsigned int distanza2 = sonar2.ping();
    distanza2=distanza2 / US_ROUNDTRIP_CM;

    Serial.print(luce1);
    Serial.print(",");
    Serial.print(luce2);
    Serial.print(",");
    Serial.print(distanza1);
    Serial.print(",");
    Serial.print(distanza2);
    Serial.print(",");
}

```

```

color = readColor();
Serial.print(scale.get_units(10)*28.3495231, 1);
Serial.println(" ");

color=0;
scale.power_down();
delay(5000);
scale.power_up();

char a;
a = Serial.read ();
switch (a) {
    case '1': // tira su
        motoreservo.write(80);
        Serial.println("SU");
        break;
    case '2': // tira giu
        motoreservo.write(15);
        Serial.println("GIU");
        break;
    case '3': // Ricalcola
        motoreservo.write(60);
        delay(1000);
        motoreservo.write(70);
        Serial.println("Ricalcola");
        break;
    default : // In tutti gli altri casi visualizzo un messaggio
        break;
}

}

```

```

//Read-Color Function
int readColor() {

    //Setting red filtered photodiodes to be read
    digitalWrite(s2, LOW);
    digitalWrite(s3, LOW);

    //Reading the output frequency
    frequency = pulseIn(out, LOW);
    int R = frequency;

    //Printing the value on the serial monitor
    Serial.print(frequency); //printing rosso color frequency
    Serial.print(",");
    Serial.print(" ");

    //Setting Green filtered photodiodes to be read
    digitalWrite(s2, HIGH);
    digitalWrite(s3, HIGH);

    //Reading the output frequency
    frequency = pulseIn(out, LOW);
    int G = frequency;

    //Printing the value on the serial monitor
    Serial.print(frequency);
    //printing verde color frequency
    Serial.print(",");
    Serial.print(" ");

    //Setting Blue filtered photodiodes to be read
    digitalWrite(s2, LOW);
    digitalWrite(s3, HIGH);

    //Reading the output frequency
    frequency = pulseIn(out, LOW);
    int B = frequency;

    //Printing the value on the serial monitor
    Serial.print(frequency); //printing blu color frequency
}

```

```

Serial.print(",");
if(R<260 & R>230 & G<860 & G>800){
    color = 1; // Rosso
}
if(G<420 & G>370 & B<350 & B>305){
    color = 2; // Blu
}
if(R<450 & R>420 & G<420 & G>390){
    color = 3; // Verde
}
return color;
}

```

---

#### A.1.4 Codice software per salvare i dati (VB.NET)

##### Form1: Funzione

---

```

Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs)
        Handles Button1.Click
        Form2.Show()
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs)
        Handles Button2.Click
        Form3.Show()
    End Sub
End Class

```

---

##### Form1: Grafica

---

```

<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()>
-
Partial Class Form1

```

```

Inherits System.Windows.Forms.Form

'Form esegue l'override del metodo Dispose per pulire l'elenco
dei componenti.
<System.Diagnostics.DebuggerNonUserCode()> _
Protected Overrides Sub Dispose(ByVal disposing As Boolean)
    Try
        If disposing AndAlso components IsNot Nothing Then
            components.Dispose()
    End If
    Finally
        MyBase.Dispose(disposing)
    End Try
End Sub

'Richiesto da Progettazione Windows Form
Private components As System.ComponentModel.IContainer

'NOTA: la procedura che segue è richiesta da Progettazione
Windows Form
'Può essere modificata in Progettazione Windows Form.
'Non modificarla mediante l'editor del codice.
<System.Diagnostics.DebuggerStepThrough()> _
Private Sub InitializeComponent()
    Me.Button1 = New System.Windows.Forms.Button()
    Me.Button2 = New System.Windows.Forms.Button()
    Me.SuspendLayout()
    '
    'Button1
    '
    Me.Button1.Location = New System.Drawing.Point(1, 1)
    Me.Button1.Name = "Button1"
    Me.Button1.Size = New System.Drawing.Size(456, 297)
    Me.Button1.TabIndex = 0
    Me.Button1.Text = "Addestramento"
    Me.Button1.UseVisualStyleBackColor = True
    '
    'Button2
    '
    Me.Button2.Location = New System.Drawing.Point(530, 1)

```

```

Me.Button2.Name = "Button2"
Me.Button2.Size = New System.Drawing.Size(456, 297)
Me.Button2.TabIndex = 1
Me.Button2.Text = "Riconoscimento"
Me.Button2.UseVisualStyleBackColor = True
,
'Form1
,
Me.AutoScaleDimensions = New System.Drawing.SizeF(16.0!,
31.0!)
Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font
Me.ClientSize = New System.Drawing.Size(996, 312)
Me.Controls.Add(Me.Button2)
Me.Controls.Add(Me.Button1)
Me.Name = "Form1"
Me.Text = "Form1"
Me.ResumeLayout(False)

End Sub

Friend WithEvents Button1 As Button
Friend WithEvents Button2 As Button
End Class

```

---

**Form2**

---

```

num(0).
num(s(X)) :- num(X).

```

---

## A.2 Addestramento dei modelli

### A.2.1 Codice modelli (Python)

---

```

num(0).
num(s(X)) :- num(X).

```

---

## A.3 Applicazioni

### A.3.1 Codice software per differenziare (VB.NET)

Form3

---

```
num(0).  
num(s(X)) :- num(X).
```

---

### A.3.2 Codice Bot Telegram per differenziare (Python)

---

```
num(0).  
num(s(X)) :- num(X).
```

---

# Bibliografia

- [1] Duccio Bianchi (2019), L'Economia Circolare in Italia: la filiera del riciclo asse portante di un'economia senza rifiuti, Edizioni Ambiente.
- [2] Mancini Giovanna (2019), Economia circolare: il riciclo del legno vale 1,4 miliardi e 6mila posti di lavoro, Il Sole 24 Ore.
- [3] Simon Munzert - Christian Rubba - Peter Meibner - Dominic Nyhuis (2014), Automated Data Collection with R: A Practical Guide to Web Scraping and Text Mining, John Wiley & Sons Inc.
- [4] Paolo Guidi (2018), Elementi di Domotica, Zanichelli.
- [5] Paolo Guidi (2017), Fondamenti di robotica, Zanichelli.
- [6] Luigi Lo Russo - Elena Bianchi (2016), Arduino, Hoepli.
- [7] Paolo Di Leo (2017), Sensori & Arduino, Libri Sandit.