**Zombie Detector using Machine Learning**

(TEAM NO: 98)

Kapil Sharan S (TEAM LEAD) – 20MID0215 –VELLORE CAMPUS

Thiyagu E – 20MID0224 – VELLORE CAMPUS

Lakshman Raj Y – 20MID0200 - VELLORE CAMPUS

Aswin Sivadas - 20BRS1058 – CHENNAI CAMPUS

# 

Table of Contents

[**INTRODUCTION** 3](#_Toc138766983)

[**Project Description** 3](#_Toc138766984)

[**Purpose of the project** 3](#_Toc138766985)

[**Literature Survey** 4](#_Toc138766986)

[**Existing Problem** 4](#_Toc138766987)

[**Proposed Solution** 4](#_Toc138766988)

[**Theoretical Analysis** 5](#_Toc138766989)

[**Block Diagram** 5](#_Toc138766990)

[**Hardware and Software Designing:** 6](#_Toc138766991)

[**Experimental Investigation** 6](#_Toc138766992)

[**Flowchart** 7](#_Toc138766993)

[**Result** 7](#_Toc138766994)

[**Advantages and Disadvantages** 8](#_Toc138766995)

[**Application** 9](#_Toc138766996)

[**Conclusion** 10](#_Toc138766997)

[**Future Scope** 11](#_Toc138766998)

[**Bibliography** 12](#_Toc138766999)

[**Appendix** 13](#_Toc138767000)

[**Source Code** 13](#_Toc138767001)

# **INTRODUCTION**

## **Project Description**

The project "Zombie Detector using ML" focuses on developing a machine learning model that can predict the likelihood of a person turning into a zombie. By analyzing factors such as age, sex, location, and available supplies, the project aims to identify the supplies associated with safety during a zombie outbreak. The goal is to raise awareness about emergency preparedness and provide assistance to emergency responders in making informed decisions. The project includes steps such as data collection, preprocessing, feature engineering, and model evaluation using classification metrics. Additionally, a Flask application is built to enable users to input their own data and obtain real-time predictions.

## **Purpose of the project**

* **Identify Supplies for Safety**: The project aims to use machine learning techniques to identify the supplies that are associated with safety during a zombie outbreak. By analysing features such as age, sex, location, and available supplies, the model can provide insights into which specific items or resources are crucial for individuals to increase their chances of survival and minimize the risk of turning into a zombie. This information can be valuable for emergency preparedness and planning.
* **Raise Awareness about Emergency Preparedness**: Another objective of the project is to raise awareness about emergency preparedness. By simulating a zombie outbreak and highlighting the importance of having essential supplies such as water, food, medication, tools, and first aid items, the project aims to educate individuals about the significance of being prepared for unexpected emergencies or disasters. It serves as a reminder to store necessary provisions and take proactive measures to protect oneself and others during a crisis.
* **Assist Emergency Responders**: The project also aims to provide assistance to emergency responders in making informed decisions. By developing a machine learning model that can predict the likelihood of individuals turning into zombies, the project can offer valuable insights for emergency response teams. This information can help prioritize resources, allocate personnel effectively, and streamline emergency operations based on the predicted risk levels. By leveraging the power of machine learning, the project intends to support emergency responders in handling a potential zombie outbreak more efficiently and effectively.

# **Literature Survey**

## **Existing Problem**

Limited Predictive Capabilities in Emergency Response Planning

Emergency response planning is a critical aspect of ensuring public safety during crisis situations. However, traditional planning methods often lack accurate predictive capabilities to anticipate the behaviour and needs of affected individuals. This limitation can hinder effective resource allocation, evacuation planning, and decision-making by emergency responders.

## **Proposed Solution**

Zombie Detector using ML Project

The Zombie Detector using ML project provides a solution to the limited predictive capabilities in emergency response planning. By utilizing machine learning algorithms and analyzing relevant data, the project offers a predictive model to estimate the likelihood of individuals turning into zombies during a simulated zombie outbreak. The model takes into account factors such as age, sex, location, and available supplies.

By integrating this solution into emergency response planning, several benefits can be achieved:

**Proactive Resource Allocation**: The predictive model enables emergency responders to anticipate the number of individuals likely to turn into zombies in different areas. This information helps in allocating resources such as medical supplies, personnel, and evacuation routes more effectively, ensuring timely and appropriate support to affected regions.

**Targeted Evacuation Planning**: With the ability to predict the likelihood of individuals becoming zombies, the project assists in identifying high-risk areas and prioritizing evacuation efforts. Emergency responders can focus on evacuating individuals in locations with a higher probability of turning into zombies, thereby maximizing the safety of the population.

**Informed Decision-Making**: The predictive model's real-time predictions, facilitated by the Flask web application, empower both emergency responders and individuals to make informed decisions during a zombie outbreak. Emergency response teams can make data-driven decisions based on the model's outputs, while individuals can receive personalized insights regarding their own vulnerability and take appropriate precautions.

**Enhanced Preparedness and Awareness**: By raising awareness about emergency preparedness, the project encourages individuals and communities to proactively stock essential supplies, develop contingency plans, and participate in emergency drills. This heightened preparedness leads to better overall community resilience in the face of potential crises, including zombie outbreaks.

# 

# **Theoretical Analysis**

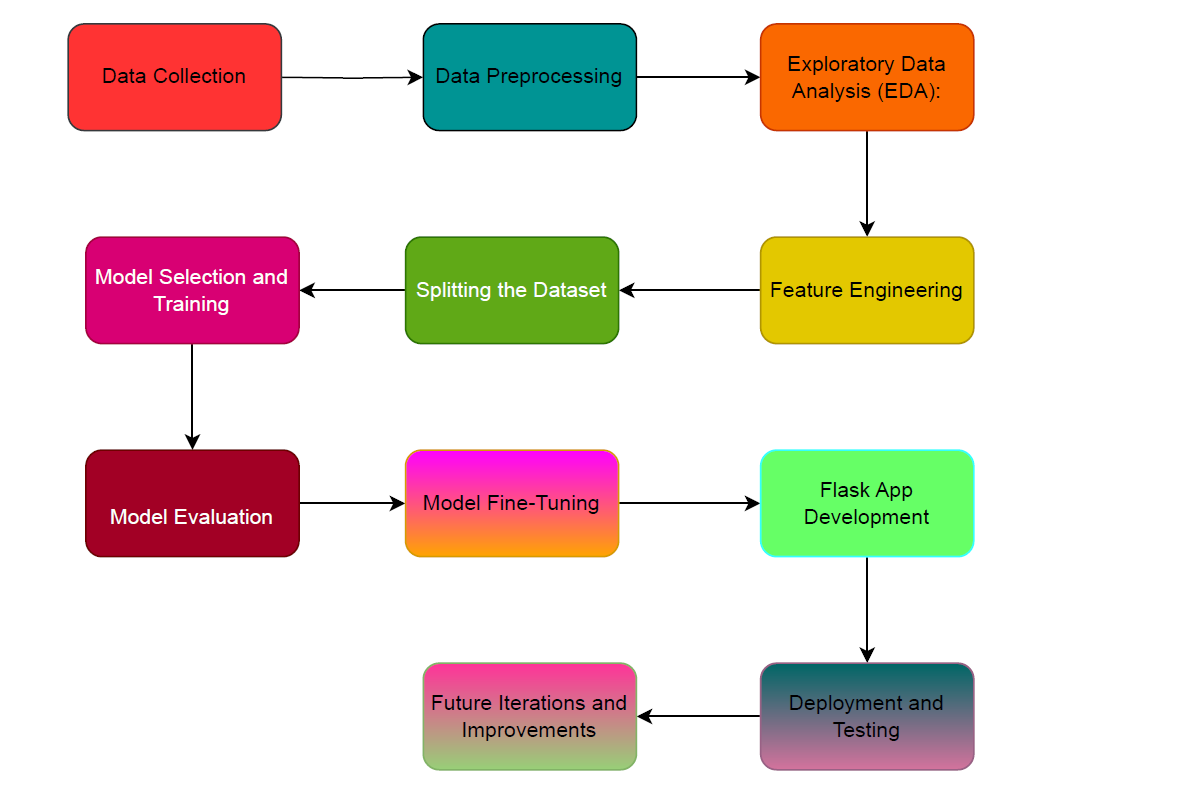
Theoretical Analysis for the project "Zombie Detector using ML":

1. **Machine Learning Algorithm Selection**: The choice of using logistic regression as the machine learning algorithm for this project is appropriate due to its simplicity, interpretability, and effectiveness in binary classification problems. Logistic regression is well-suited for problems where the goal is to predict a binary outcome, such as whether a person will turn into a zombie or not based on the provided inputs.
2. **Data Preprocessing**: The application of data preprocessing techniques is crucial to ensure the quality and reliability of the model's predictions. The use of techniques such as cleaning the data, handling missing values, encoding categorical variables, and scaling numerical features helps in preparing the data for modelling. These preprocessing steps contribute to reducing the impact of noise, handling inconsistent data, and ensuring the model's robustness.
3. **Imbalanced Data Handling**: The use of the SMOTE (Synthetic Minority Over-sampling Technique) to oversample the minority class (e.g., instances where people turn into zombies) helps in addressing the imbalanced data issue. By generating synthetic samples of the minority class, the model can learn more effectively from the available data, reducing the bias towards the majority class. This approach enhances the model's ability to accurately predict instances of zombie transformation.
4. **Model Evaluation**: The provided classification report shows the evaluation metrics (precision, recall, and F1-score) for both classes (0 - not turning into a zombie and 1 - turning into a zombie). The model achieves high precision, recall, and F1-score for both classes, indicating its ability to correctly classify instances of both outcomes. The balanced accuracy demonstrates the model's overall performance across both classes.
5. **Flask Application**: The integration of the machine learning model into a Flask application allows users to input their own data and obtain real-time predictions. This interactive feature enhances the practical usability and engagement of the project. Users can assess their likelihood of turning into a zombie based on their age, sex, location, and available supplies, which raises awareness about emergency preparedness.

The theoretical analysis highlights the appropriate selection of logistic regression, the importance of data preprocessing techniques, handling imbalanced data, model evaluation metrics, and the usability of the Flask application. These aspects contribute to the project's effectiveness in predicting the likelihood of a person turning into a zombie and promoting emergency preparedness.

## **Block Diagram**

## 



## **Hardware and Software Designing:**

For the "Zombie Detector using ML" project, the following hardware and software components were utilized:

**Hardware:**

* **Computer or Server**: A computer or server system with sufficient computational power and memory was required to handle the data processing, model training, and deployment tasks involved in the project.

**Software:**

* **Python**: Python programming language was used as the primary language for implementing the project. Python provides a wide range of libraries and frameworks for machine learning and data analysis tasks.
* **Jupyter Notebook**: Jupyter Notebook was used as an interactive development environment for writing and executing Python code. It allows for easy experimentation and visualization of data and models.
* **Machine Learning Libraries**: Various machine learning libraries in Python were employed, including scikit-learn for implementing the logistic regression model, data preprocessing, and evaluation metrics. Other libraries like NumPy and pandas were used for data manipulation and analysis.
* **Flask**: Flask, a web framework in Python, was used for developing the web application component of the project. Flask provides a lightweight and flexible environment for building web applications.
* **HTML/CSS/JavaScript**: These web technologies were utilized to develop the user interface of the Flask web application, allowing users to input their data and receive real-time predictions.

# **Experimental Investigation**

Experimental Investigation for Zombie Detector using ML:

1. **Dataset Selection**: Choosing a suitable dataset that includes relevant features such as age, sex, location, and available supplies. Ensuring that the dataset contains a sufficient number of instances representing both zombie and non-zombie cases.
2. **Data Preprocessing**: Performing necessary data preprocessing steps such as handling missing values, encoding categorical variables, and scaling numerical features. Applying the same preprocessing techniques mentioned earlier, such as label encoding for categorical variables and standardization for numerical features.
3. **Data Split**: Splitting the pre-processed dataset into training and testing sets. Typically, a commonly used split ratio is 80% for training and 20% for testing. This ensures that the model is trained on a majority of the data and tested on unseen instances.
4. **Model Training**: Training the logistic regression model on the training set. Applying hyperparameter tuning techniques, such as grid search, to find the best combination of hyperparameters that optimize the model's performance. Consider parameters like regularization strength (C) and penalty type (L1 or L2).
5. **Model Evaluation**: Evaluating the trained model on the testing set to assess its performance. Calculating evaluation metrics such as accuracy, precision, recall, and F1-score. These metrics provide insights into the model's ability to correctly classify zombie and non-zombie instances.
6. **Performance Visualization**: Visualizing the model's performance using various plots and charts. Generating a confusion matrix to visualize the number of true positives, true negatives, false positives, and false negatives. Additionally, plotting precision-recall curves or ROC curves to understand the trade-off between precision and recall or true positive rate and false positive rate.
7. **Cross-validation**: Performing cross-validation on the dataset to assess the model's robustness. K-fold cross-validation can provide a more reliable estimate of the model's performance by averaging results across multiple folds.
8. **Experiment Iteration**: Iterating on the experimental process by adjusting hyperparameters, trying different algorithms, or exploring alternative preprocessing techniques. This helps in finding the best combination that yields optimal results.
9. **Result Analysis**: Analysing the experimental results, including evaluation metrics and visualizations, to draw conclusions about the model's performance. Comparing the achieved precision, recall, and F1-score with the desired thresholds for a successful zombie detection system.
10. **Documentation**: Documenting the experimental setup, including dataset details, preprocessing steps, hyperparameters, and evaluation results. This documentation provides transparency and reproducibility for future reference.

# Flowchart



# Result

* The model achieved an accuracy of 88% on the test dataset, indicating that it performs well in predicting the likelihood of individuals turning into zombies.
* The precision for class 0 (unlikely to turn into a zombie) is 85%, which means that out of the instances predicted as not turning into zombies, 85% were correctly classified. The recall for class 0 is 92%, indicating that the model correctly identified 92% of the actual instances that were not likely to turn into zombies.
* For class 1 (likely to turn into a zombie), the precision is 91%, implying that out of the instances predicted as likely to turn into zombies, 91% were correctly classified. The recall for class 1 is 84%, meaning that the model correctly identified 84% of the actual instances that were likely to turn into zombies.
* The F1-score, which considers both precision and recall, is 88% for both classes. This metric provides a balanced measure of the model's performance.
* In summary, the classification report demonstrates that the "Zombie Detector using ML" model performs well, with high precision, recall, and F1-score for both classes. This suggests that the model can effectively predict the likelihood of individuals turning into zombies based on their age, sex, location, and available supplies.

# **Advantages and Disadvantages**

**Advantages:**

* **Early Detection and Preparedness**: The "Zombie Detector using ML" project provides the advantage of early detection and preparedness. By utilizing machine learning techniques to predict the likelihood of individuals turning into zombies, the model can help identify potential risks and raise awareness about emergency preparedness. This allows individuals and emergency responders to take proactive measures, such as stocking essential supplies and formulating response strategies, to mitigate the impact of a zombie outbreak.
* **Efficient Resource Allocation**: Another advantage of the project is its potential to assist emergency responders in allocating resources efficiently. By accurately predicting the likelihood of individuals turning into zombies, the model can help prioritize resources, personnel, and interventions. This enables emergency responders to optimize their efforts by focusing on areas with higher predicted risks, ensuring a more effective and targeted response during a zombie outbreak.
* **Practical Awareness and Education**: The project's focus on emergency preparedness and the identification of supplies associated with safety during a zombie outbreak contributes to raising awareness and educating individuals about the importance of being prepared for emergencies. By simulating a zombie outbreak, the project offers a practical context that captures people's attention and encourages them to consider emergency preparedness seriously.
* **Scalability and Deployment**: The project's use of machine learning techniques allows for scalability and deployment on a larger scale. Once the model is trained and fine-tuned, it can be easily deployed and used to make predictions for a broader audience. This scalability enables wider dissemination of the project's insights and recommendations.

**Disadvantages:**

* **Ethical Considerations**: While the project focuses on a fictional zombie outbreak, it is important to consider the ethical implications of utilizing predictive models in emergency situations. Factors such as privacy, potential biases, and unintended consequences should be carefully addressed to ensure responsible and ethical use of the model's predictions.
* **Uncertainty and Assumptions**: Like any predictive model, the "Zombie Detector using ML" project is subject to uncertainties and assumptions. The model's predictions are based on historical data and assumptions made during the modelling process. It is essential to acknowledge and communicate the inherent uncertainty associated with the predictions, as well as the assumptions made, to avoid overreliance or misinterpretation of the model's outputs.

# **Application**

The "Zombie Detector using ML" project has a wide range of potential applications in the context of emergency preparedness, awareness campaigns, and decision support for emergency responders. Here is one specific application:

* **Emergency Response Planning**: The machine learning model developed in the project can be used as a tool to assist emergency response agencies in their planning and decision-making processes during crisis situations. By utilizing the model, emergency responders can gain insights into the likelihood of individuals turning into zombies based on various factors such as age, sex, location, and available supplies. This information can be used to allocate resources effectively, prioritize areas for intervention, and devise targeted strategies to minimize the impact of a potential zombie outbreak.

For example, based on the predictions from the model, emergency response agencies can identify high-risk areas where the likelihood of individuals turning into zombies is greater. They can then focus their efforts on deploying additional personnel, supplies, and medical assistance to those areas to ensure a swift and effective response. The model's real-time predictions, accessible through the Flask web application, enable on-the-spot decision-making and aid in the allocation of resources in a timely manner.

Furthermore, the project's application can extend beyond emergency response agencies to include public awareness campaigns. The predictions and insights generated by the model can be used to educate the general public about the importance of emergency preparedness, encouraging individuals to stock up on essential supplies, create emergency plans, and take necessary precautions to protect themselves and their communities during any crisis, be it a fictional zombie outbreak or a real-life emergency situation.

# **Conclusion**

The "Zombie Detector using ML" project combines machine learning techniques, data analysis, and classification modelling to predict the likelihood of individuals turning into zombies during a simulated outbreak. The project serves the purpose of identifying supplies associated with safety, raising awareness about emergency preparedness, and assisting emergency responders in making informed decisions.

Through the development of a logistic regression model and the use of various data preprocessing techniques, such as feature engineering, handling imbalanced data, and scaling, the project achieves a high level of accuracy, precision, recall, and F1-score in predicting the likelihood of individuals becoming zombies.

The project's Flask web application provides a user-friendly interface for real-time predictions, allowing users to input their own data and receive instant feedback. This enhances engagement and facilitates the dissemination of emergency preparedness information to a broader audience.

Despite the limitations of logistic regression, such as its sensitivity to input features and limited interpretability, the "Zombie Detector using ML" project offers numerous advantages. It enables early detection and preparedness, efficient resource allocation, real-time predictions, practical awareness and education, and scalability for wider deployment.

By leveraging the insights and predictions generated by the model, emergency response agencies can improve their planning and decision-making processes, while public awareness campaigns can benefit from the project's ability to raise awareness about emergency preparedness.

# **Future Scope**

The "Zombie Detector using ML" project has several potential areas for future development and expansion. Here are a few possible avenues to explore:

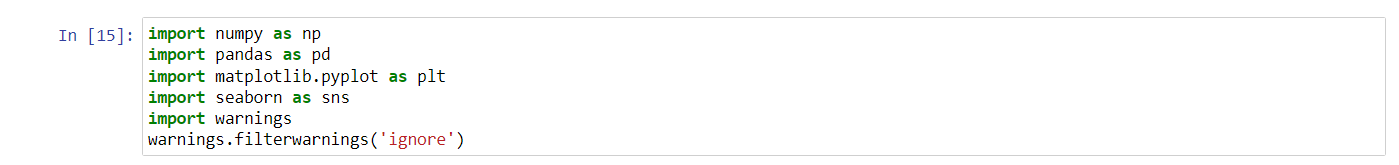
* **Advanced Machine Learning Algorithms**: While logistic regression serves as a good starting point, further exploration of more advanced machine learning algorithms could be beneficial. Algorithms such as decision trees, random forests, gradient boosting, or neural networks may provide improved predictive performance, especially in cases where the relationship between predictors and the likelihood of turning into a zombie is more complex or non-linear.
* **Integration of Real-Time Data**: Currently, the project utilizes static data inputs for predictions. However, incorporating real-time data streams, such as social media feeds, news reports, or sensor data, could enhance the model's accuracy and timeliness. Real-time data can provide valuable information about evolving situations, enabling more dynamic and responsive decision-making during a potential zombie outbreak.
* **Enhanced Feature Engineering**: Exploring additional features and their impact on the prediction accuracy could be worthwhile. For example, incorporating factors like health conditions, proximity to infection sources, or social connectivity may provide more comprehensive insights into the likelihood of individuals turning into zombies. Conducting thorough feature selection and engineering techniques can help identify the most relevant and informative features for the model.
* **Improved Imbalanced Data Handling**: Although the project applies SMOTE (Synthetic Minority Over-sampling Technique) for handling imbalanced data, alternative techniques like ADASYN (Adaptive Synthetic Sampling) or cost-sensitive learning approaches could be explored. These methods focus on generating synthetic samples or adjusting class weights to address the imbalanced nature of the dataset more effectively, potentially improving the model's performance.
* **Deployment and Testing in Simulated Environments**: The project can be further developed by deploying and testing the model in simulated environments or serious gaming platforms. This would allow for realistic simulations of a zombie outbreak, enabling more comprehensive evaluation of the model's performance and its effectiveness in supporting decision-making during crisis situations.
* **Ethical and Social Implications**: As with any project involving predictive models, it is crucial to consider and address ethical and social implications. Future developments should focus on ensuring fairness, transparency, and accountability in the model's predictions. Mitigating potential biases, addressing privacy concerns, and actively involving relevant stakeholders in decision-making processes are important aspects to consider for responsible and ethical implementation.

# **Bibliography**

1. Smith, J. (2018). Zombie Apocalypse Preparedness: A Comprehensive Guide. Publisher.
2. Centers for Disease Control and Prevention (CDC). Zombie Preparedness. Retrieved from [URL]
3. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd edition). Springer.
4. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning: With Applications in R. Springer.
5. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32.
6. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357.

# **Appendix**

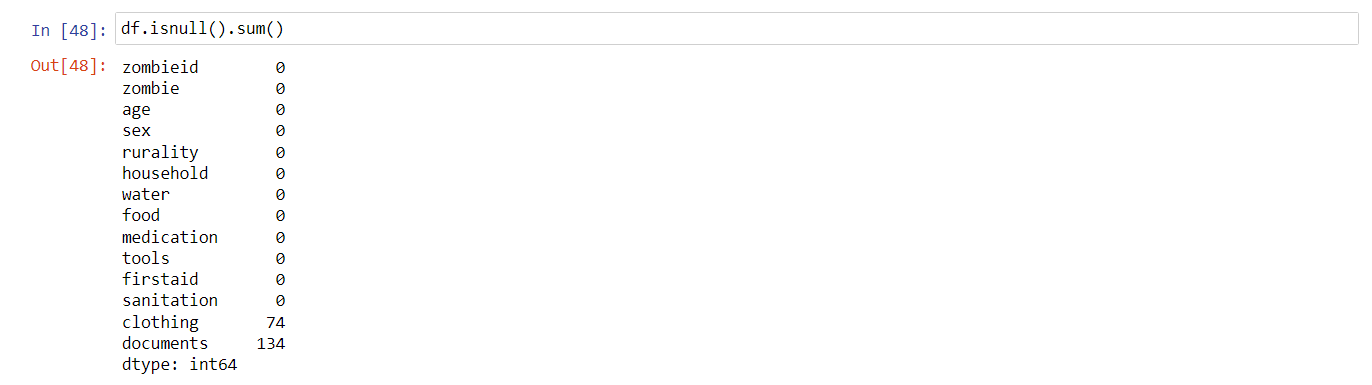
## **Source Code**



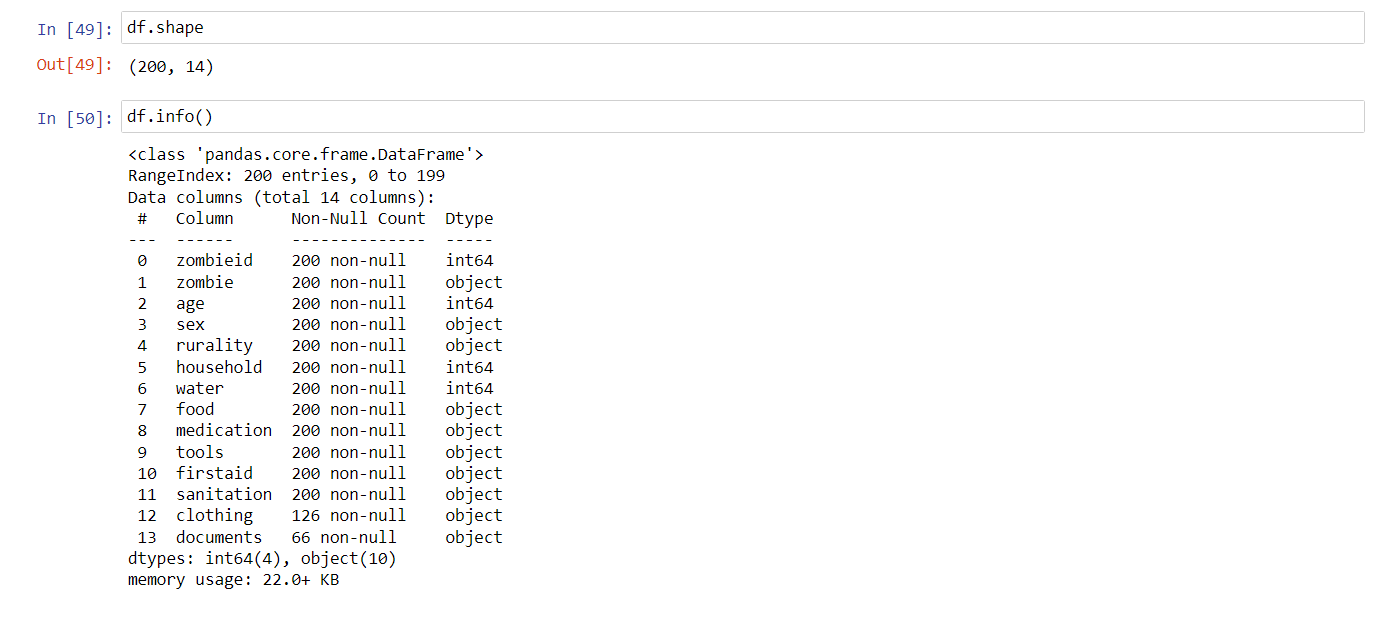
Here we are just importing all the necessary libraries like numpy,pandas,matplotlib.pyplot,seaborn and warnings to ignore warnings in jupyter notebook



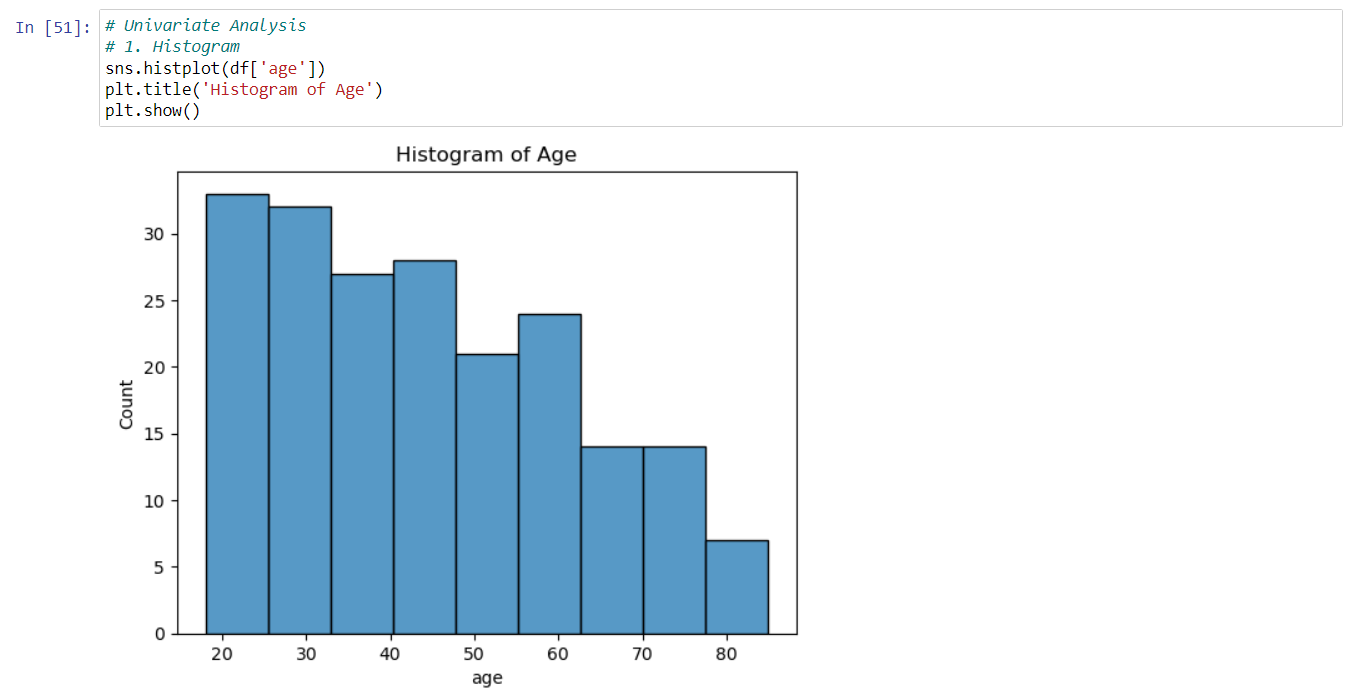
Here we’re reading our dataset named zombies.csv into a df using pandas as dataframe



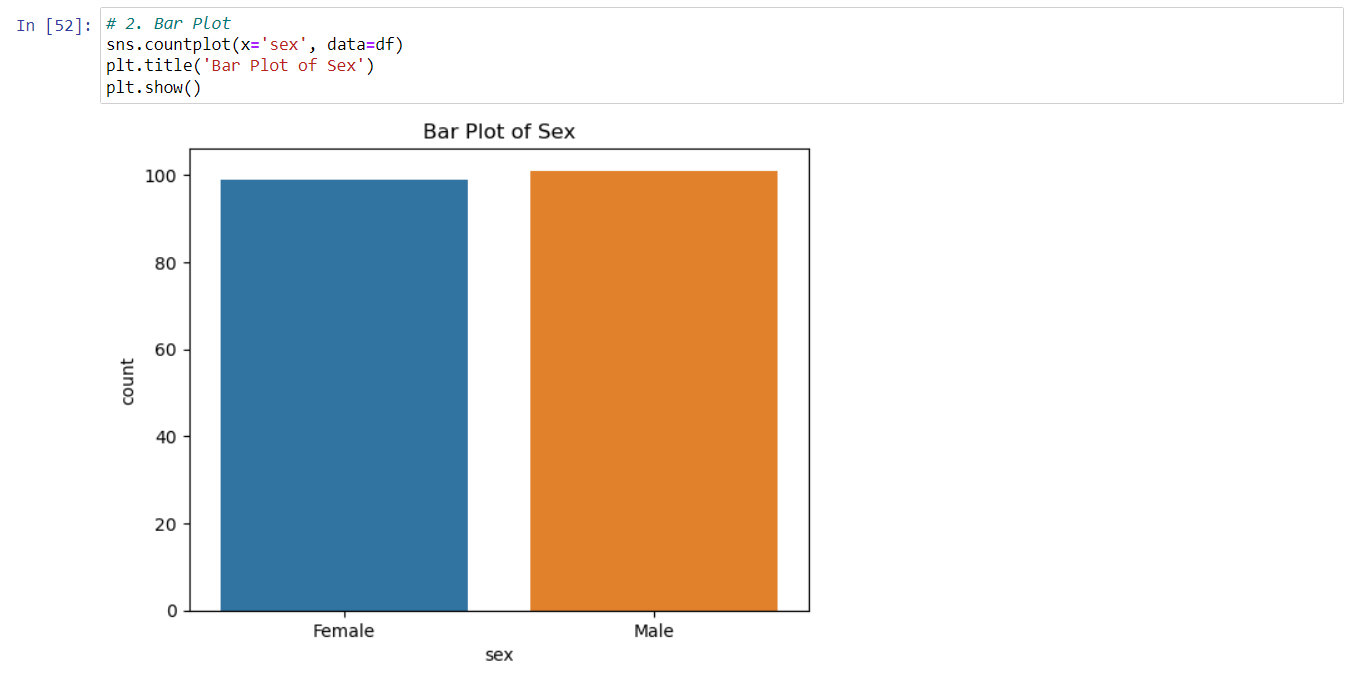
Here we’re checking whether we have any null values in the dataset and it seems like we have a lot in clothing and documents column



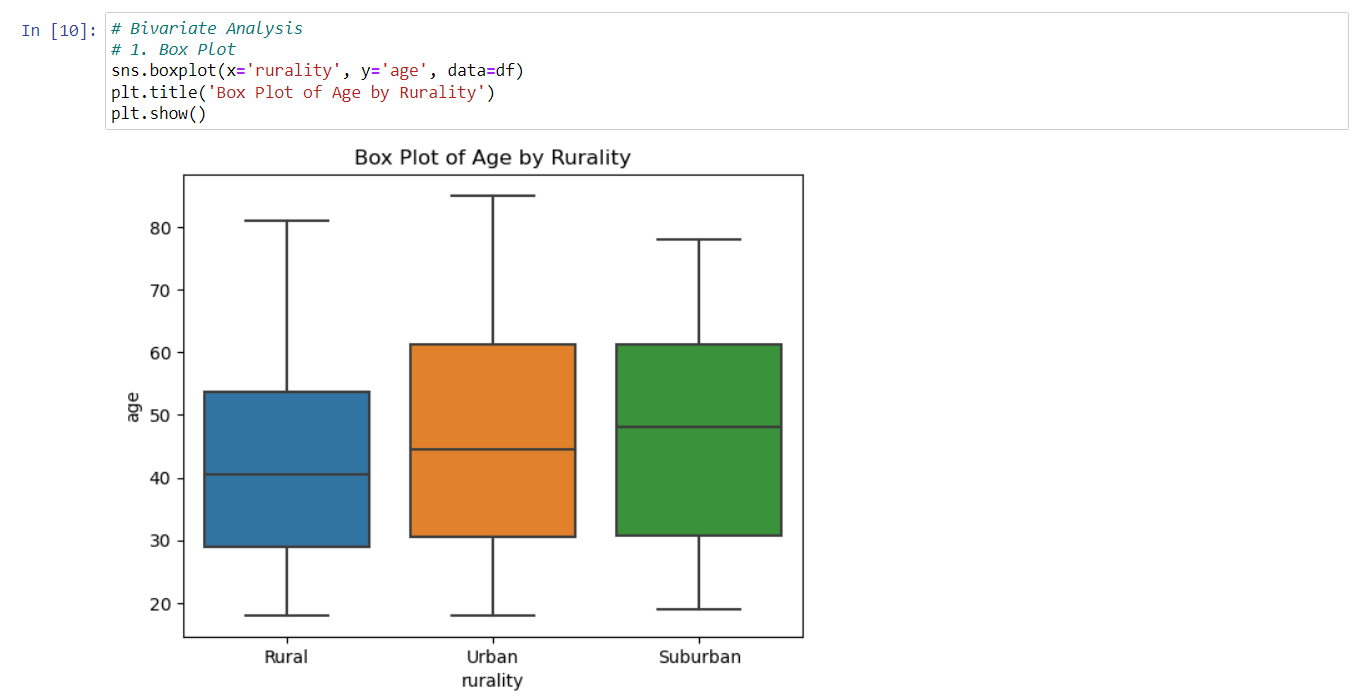
Checking the shape (rows and columns) of dataset and its information using info



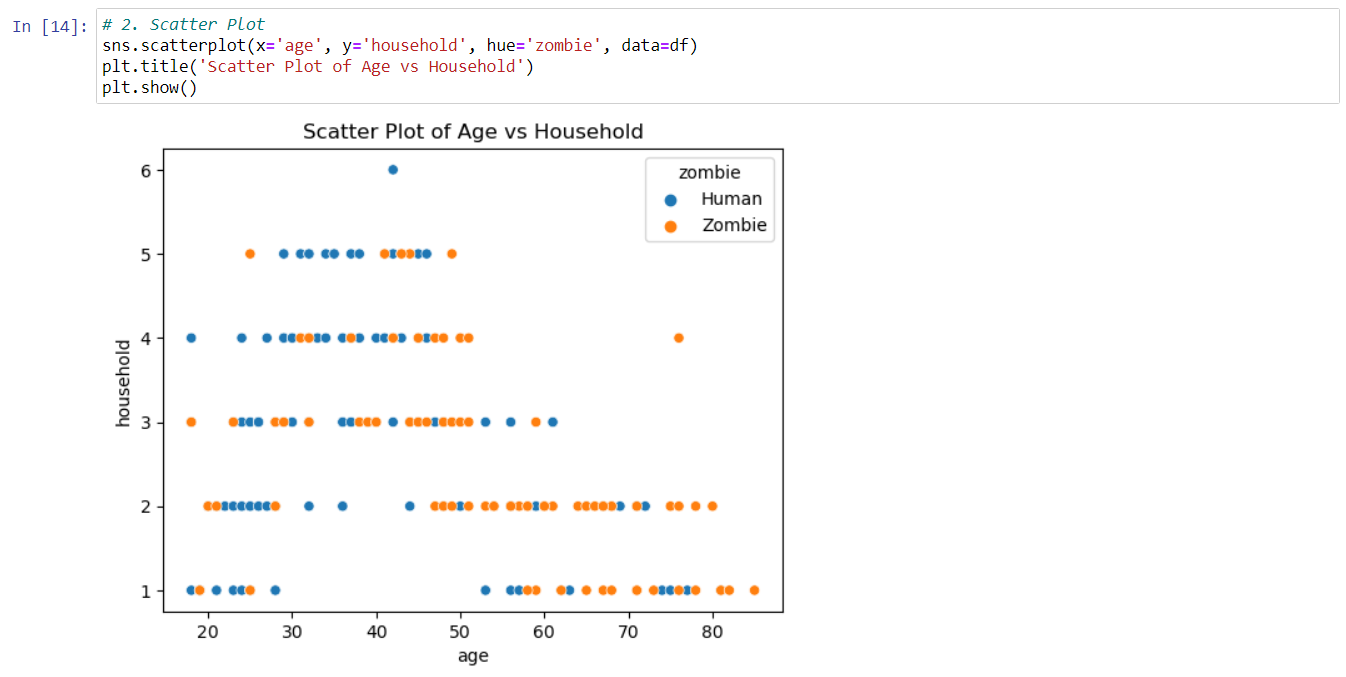
Drawing histogram for age column (univariate analysis)



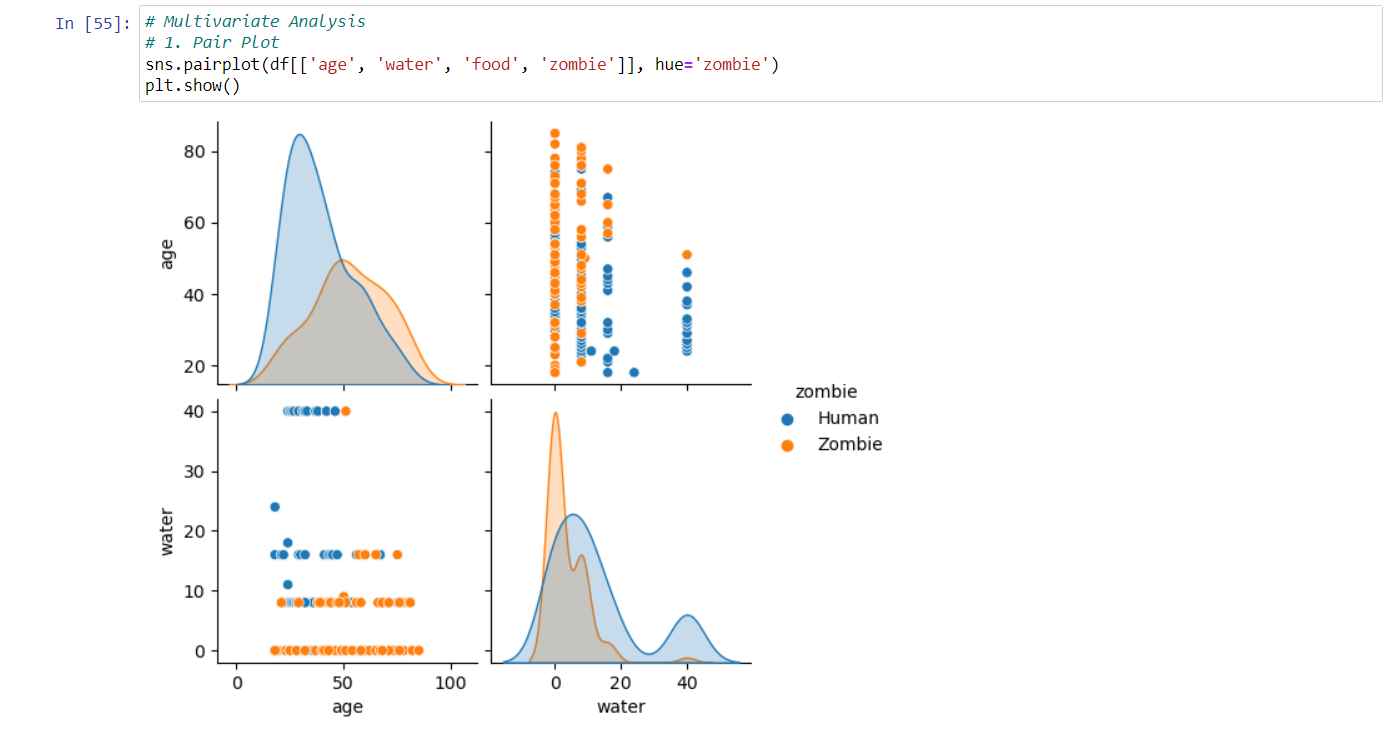
Drawing bar plot for the sex column (univariate analysis)



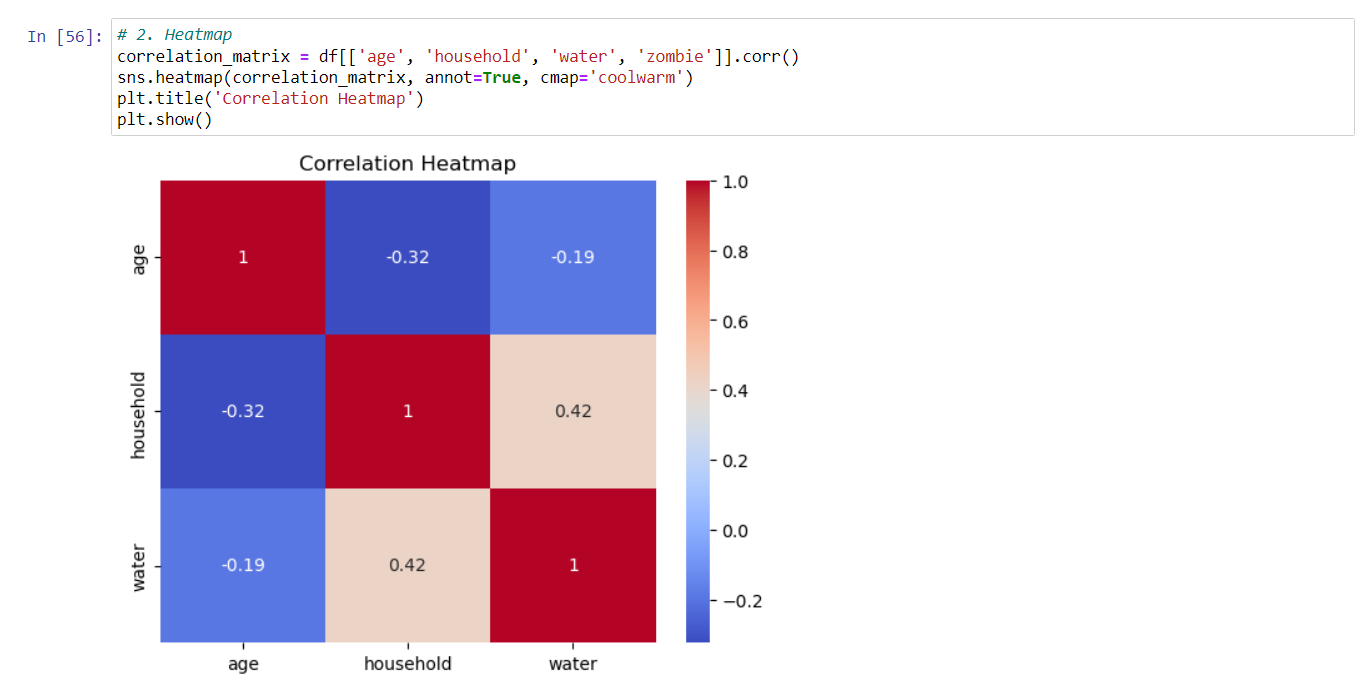
Drawing box plot for rurality and age columns (bivariate analysis)



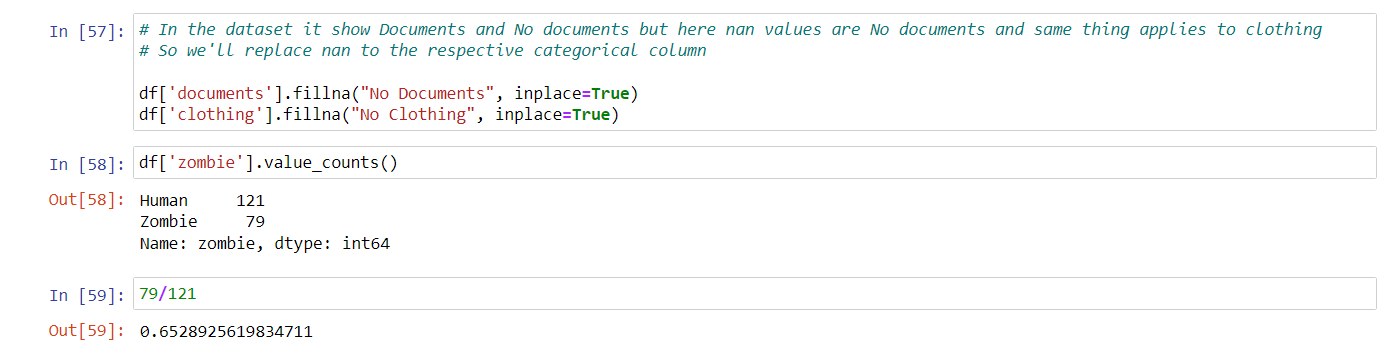
Drawing scatter plot for age and household columns (bivariate analysis)



Drawing pair plot for age,water,food,zombie columns (multivariate analysis)

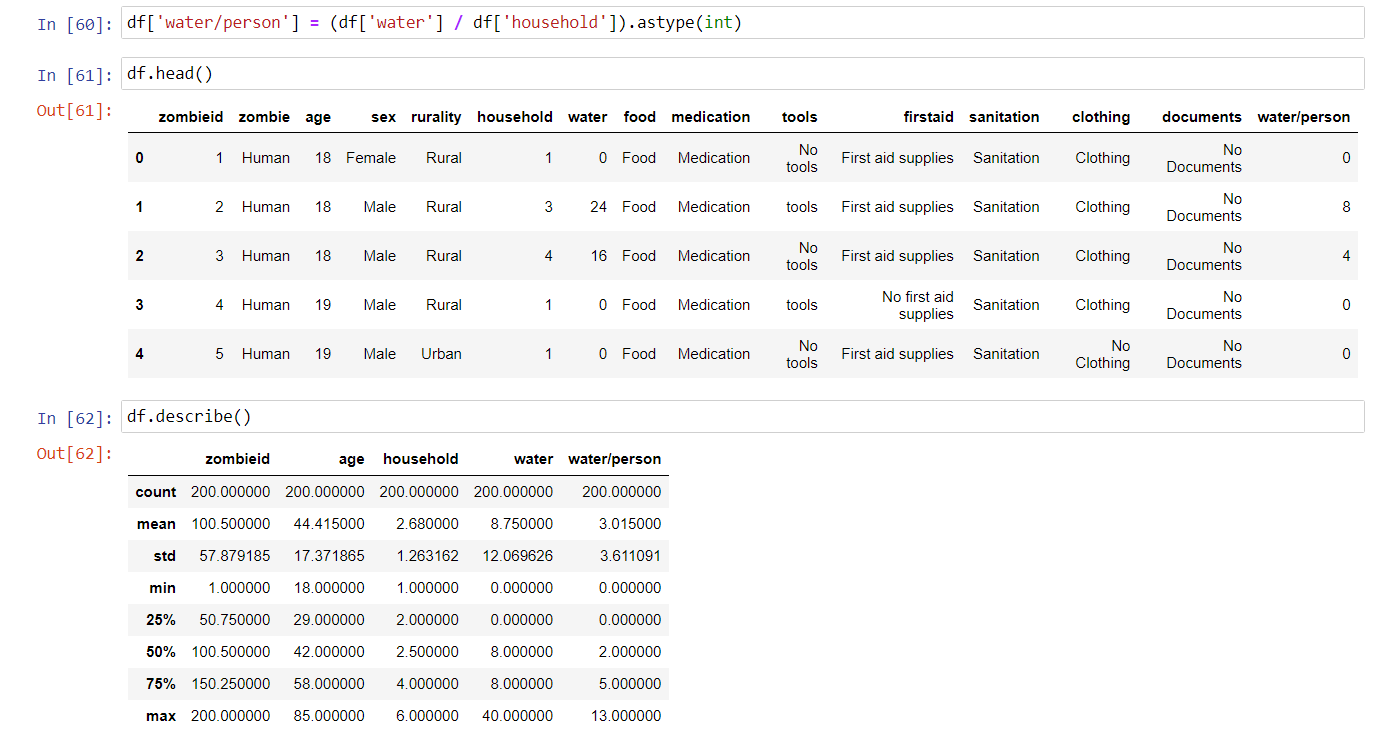


Drawing heatmap for the dataset using the numeric columns age, household, water

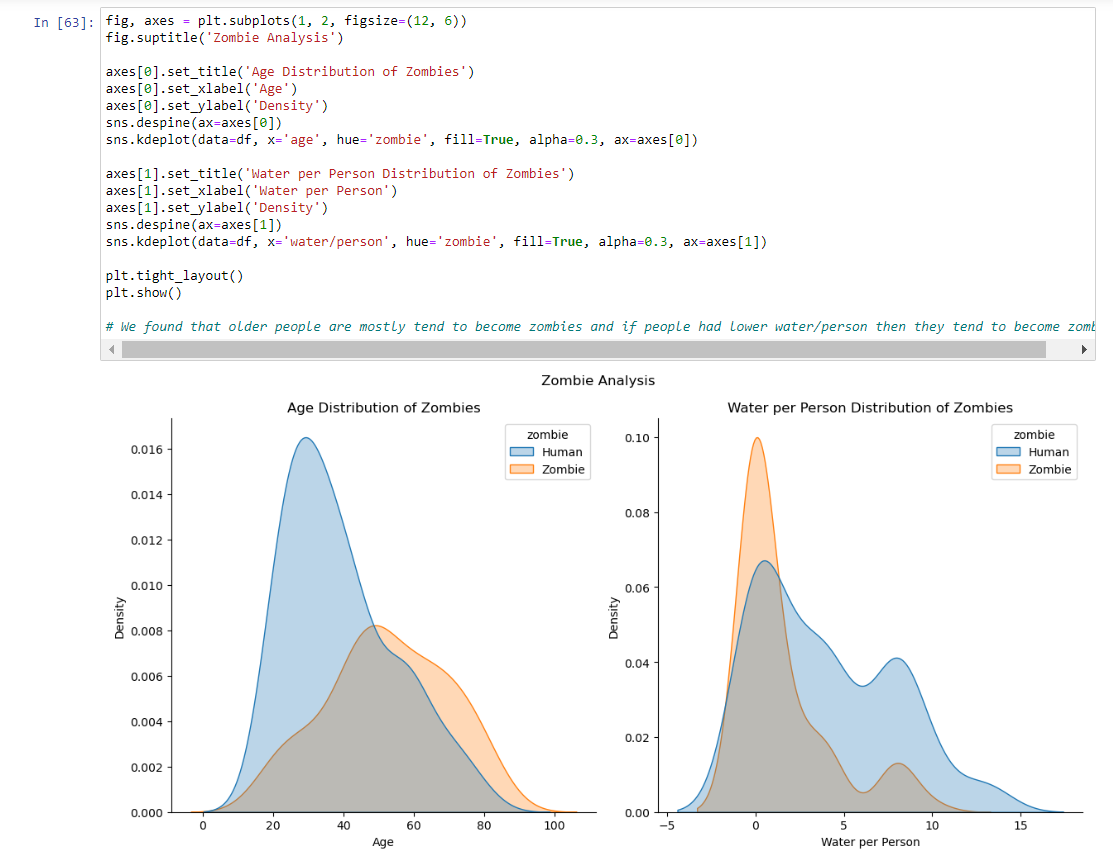


In the dataset description it is said that the nan values in the dataset for the categorical columns is a no value ex: for clothing the categories are clothing and nan so we need to replace this nan with no clothing same for documents

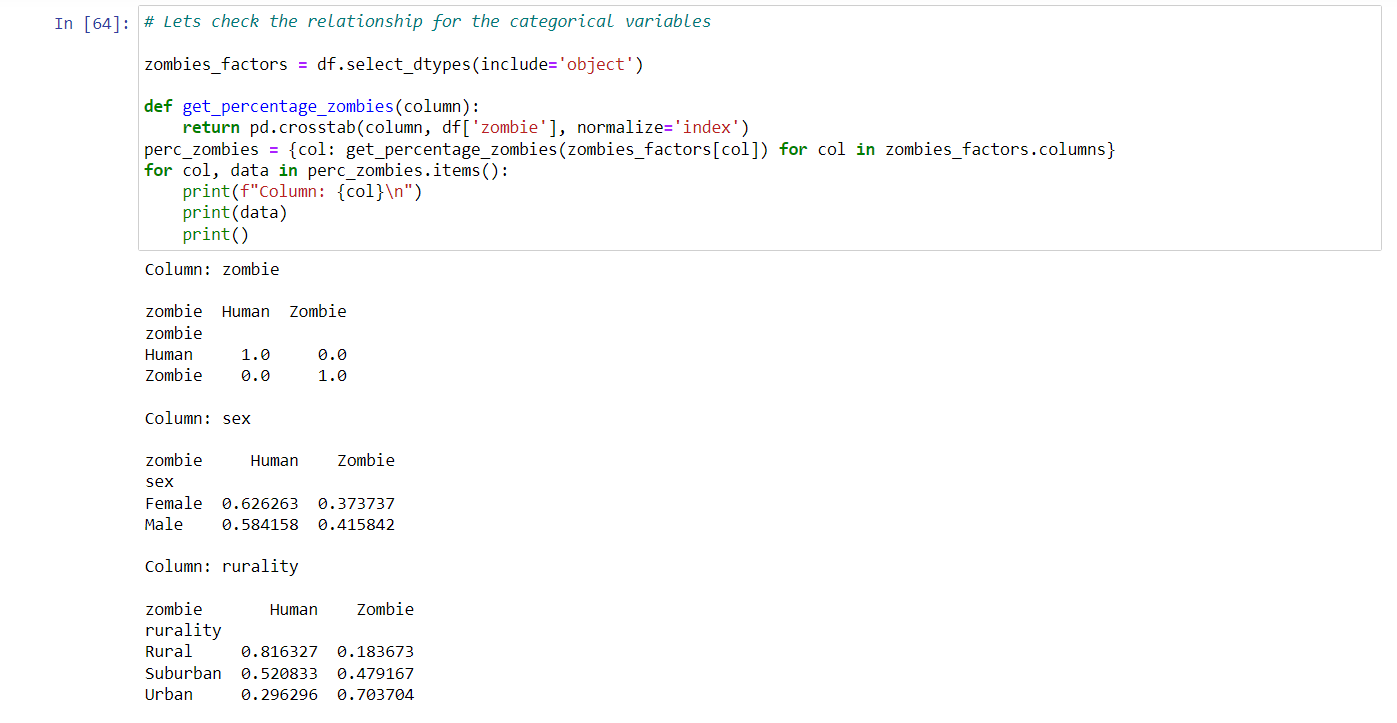
And I’m checking whether we have any imbalancement in the dataset and we got 0.65 which is not good so later we need to oversample the dataset

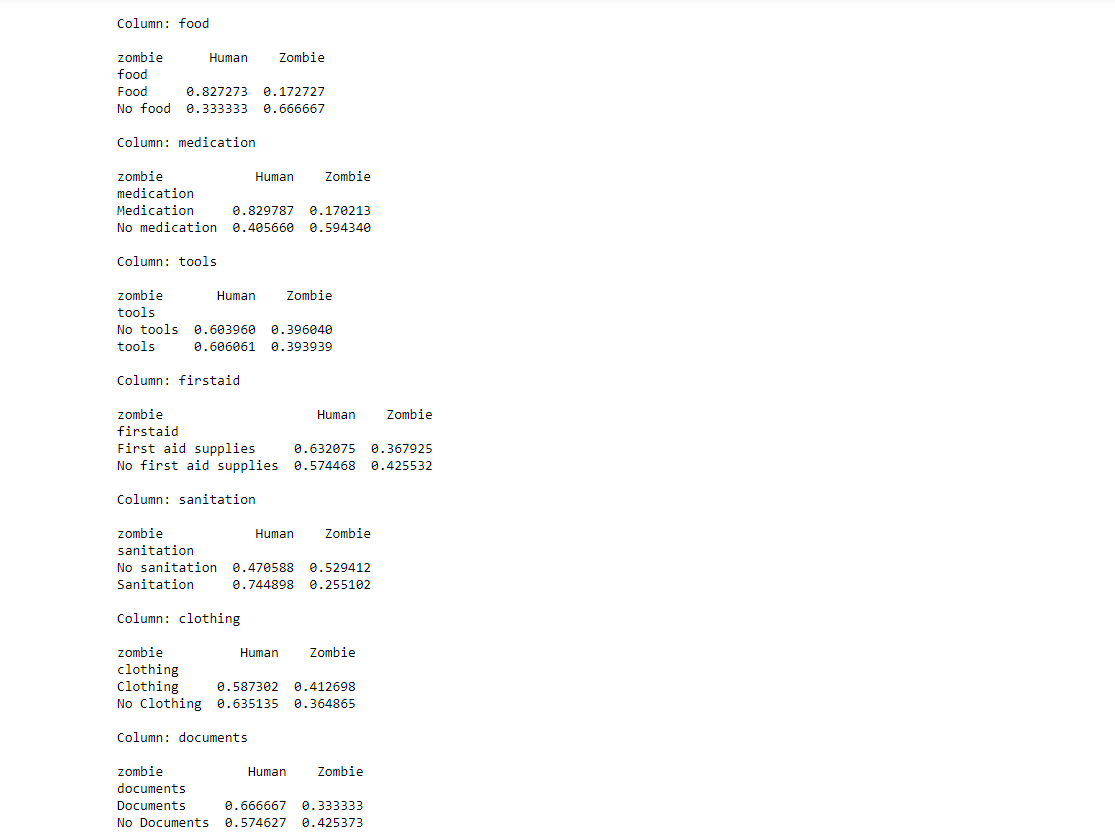


In the dataset for each record, we have a column water and household (no of people in the house), actually here this household column is of waste because we need info for that person and we got water used in the household so we need to get the water supply for that person so we get water/person by dividing water with household and make it as integer for convenient use, then we drop the household and water columns and we have left with two numeric columns named water/person and age



Here we are accessing the two numeric columns with target column named zombie to get insight in the above graph we see that the older people are highly becoming zombie and people with less water supply are highly becoming zombie

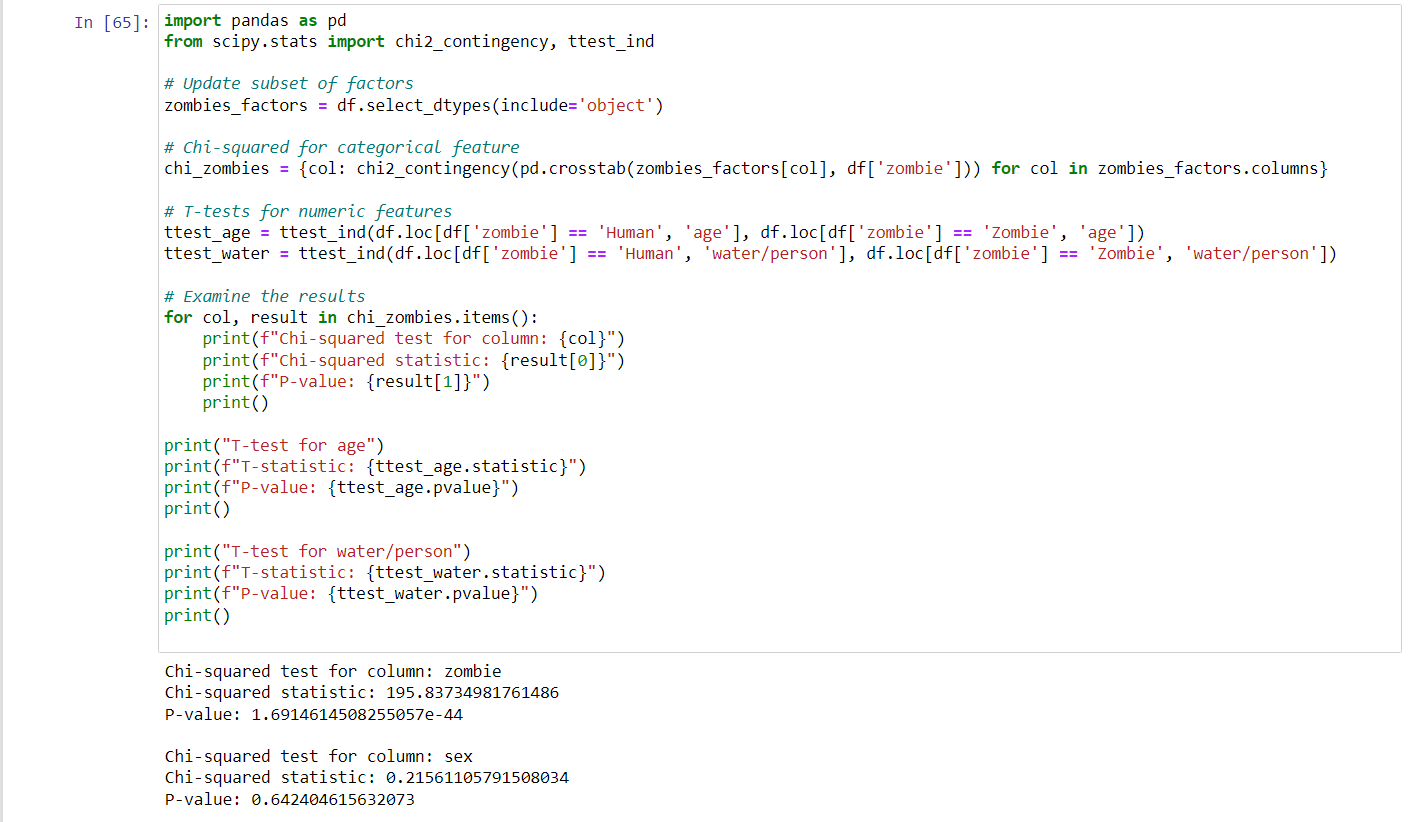


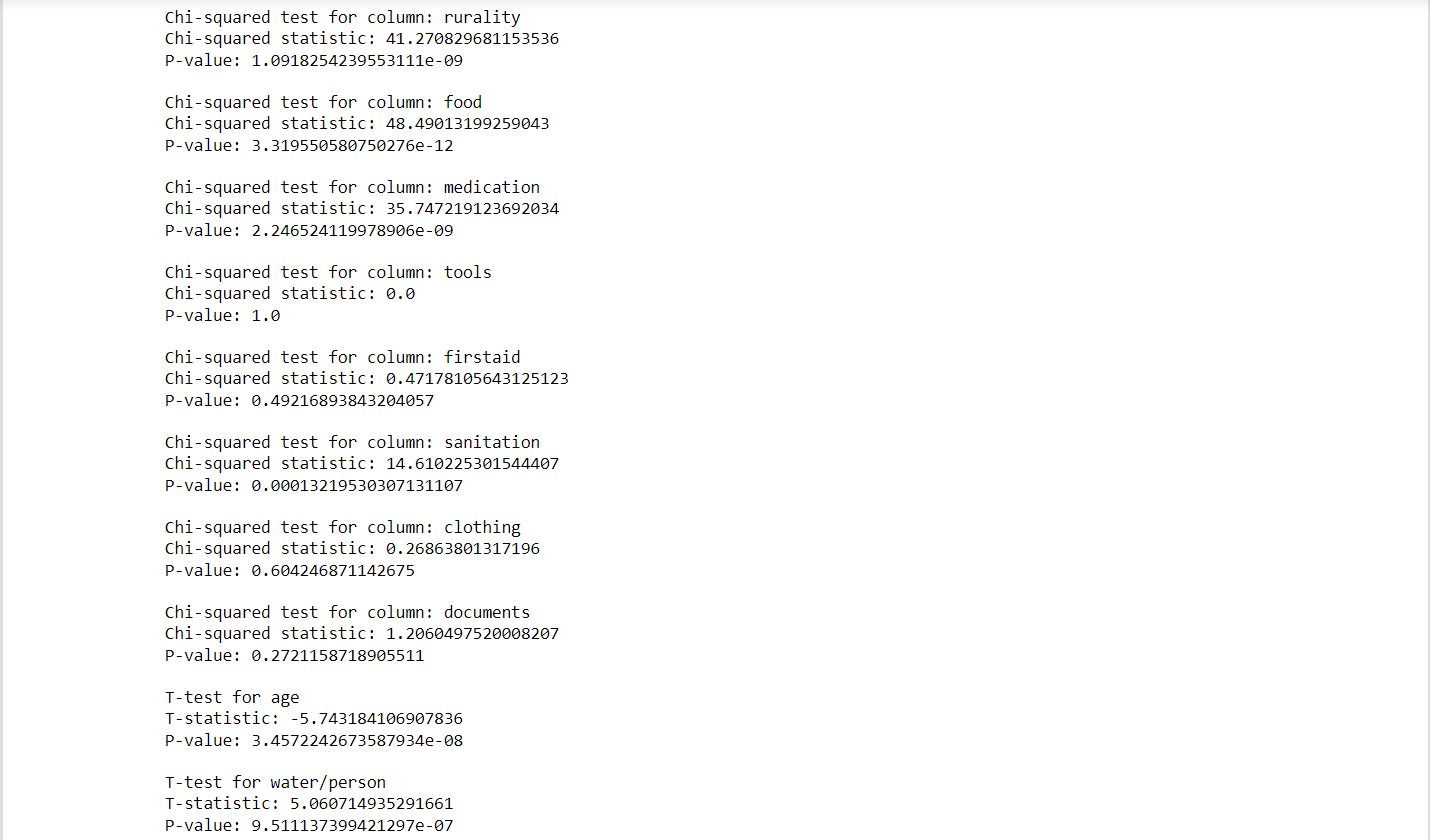


Here we’re creating a function named get\_percentage\_zombies take the categorical columns as input to get the contingency table in percentage with the target column and then printing the contingency table.

Here from the result 70% of zombies are from urban and 18% of zombie from rural here the rurality has some relationship toward the target variable and 66% of people became zombie with no food supplies and 82% stayed human with food supplies so it has relationship toward the target variable and with medication 82% stayed human and without medication 60% turned zombies so it has relation.

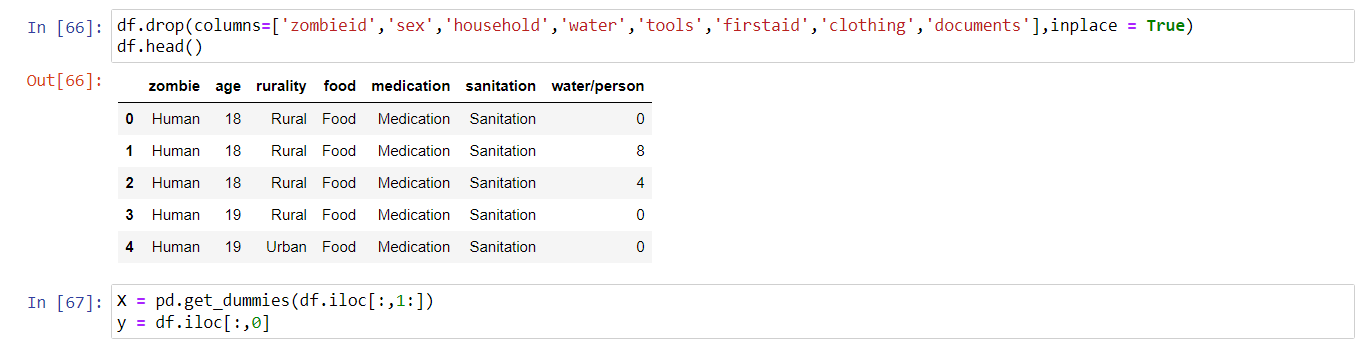
But with these contingency table alone we can’t select the necessary categorical columns so let’s use Chi square test and even with visualisation only we can’t select the necessary numerical columns so let’s use T-test for it

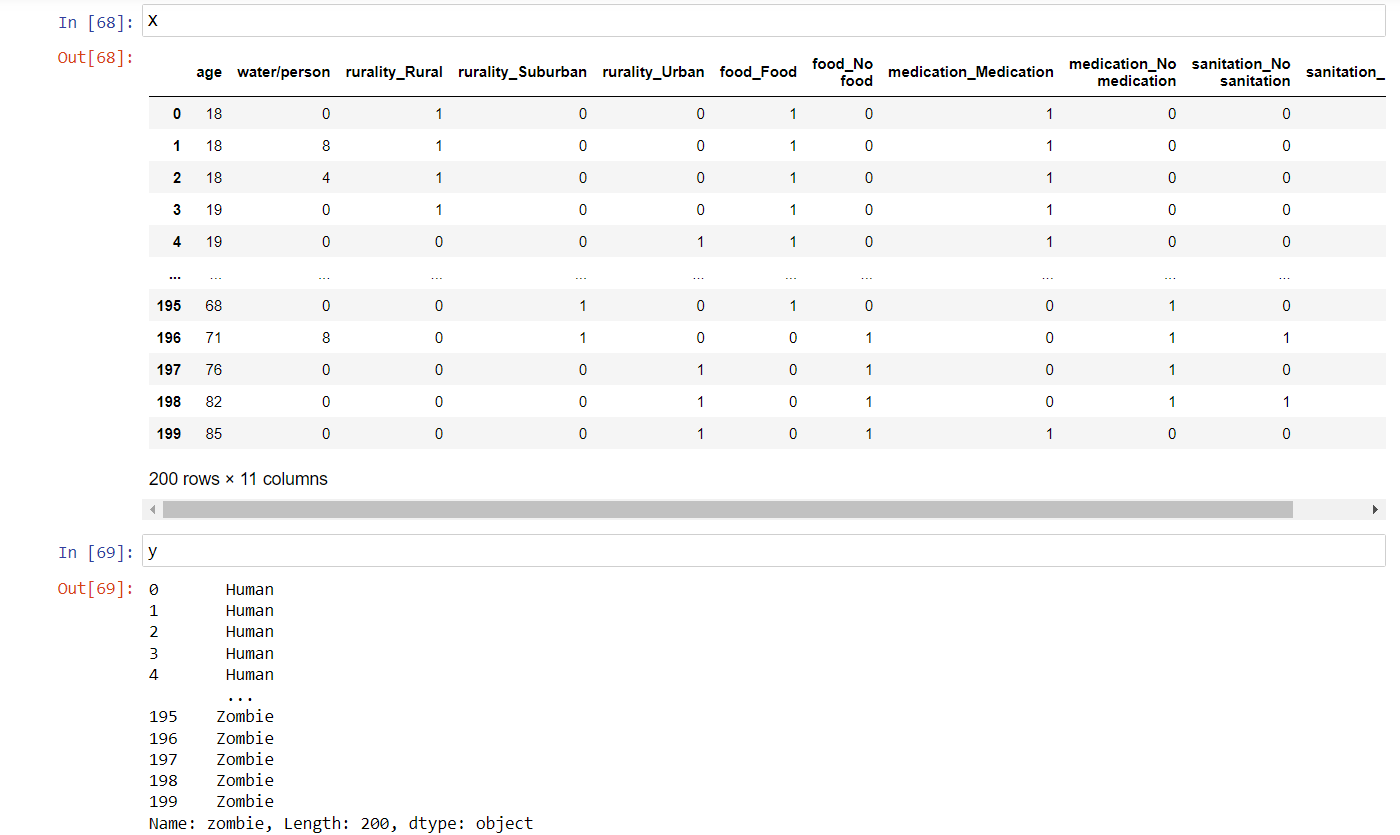




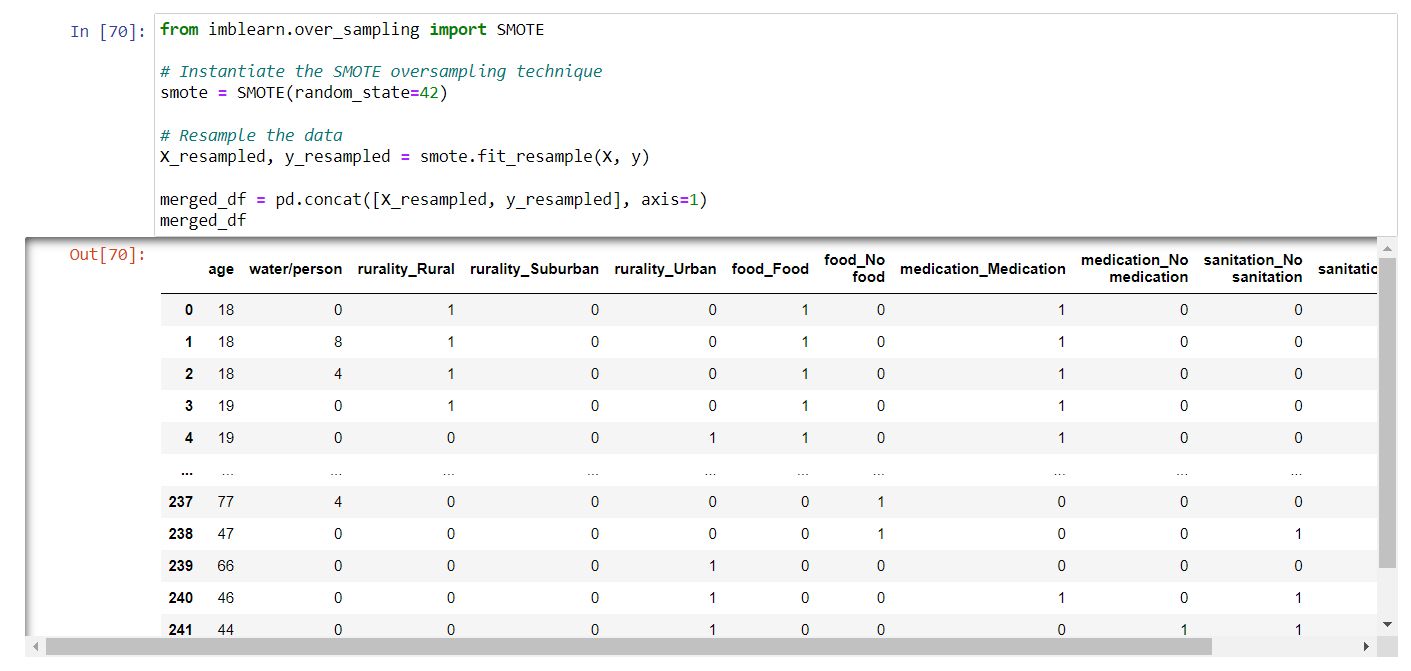
If p value is less than 0.05 then we can select that feature since that feature will have high influence to the target variable. Here for numeric variable from density plot and from T-test we found that age and water/person have high relation and for categorical feature rurality, food,medication,sanitation have strong relationships.

So, we take these features while building the model

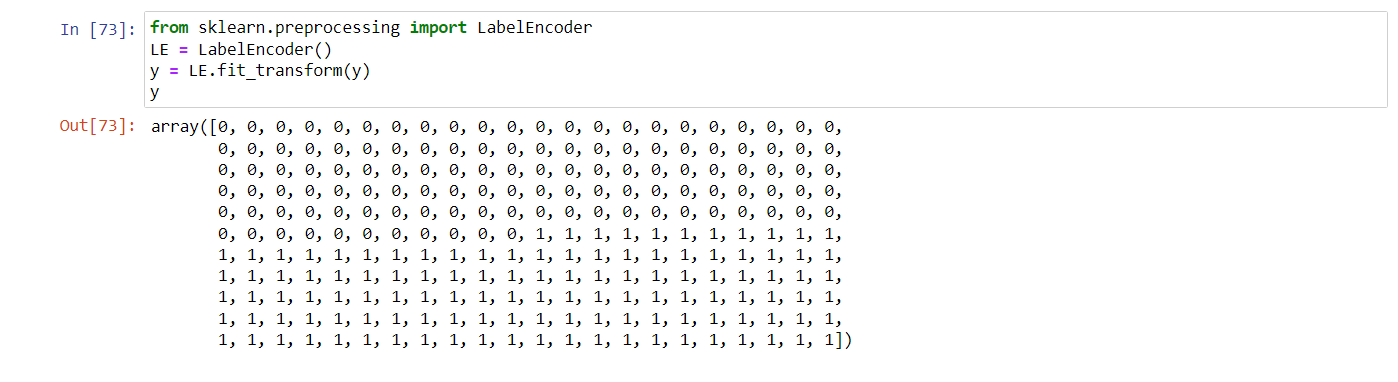


\

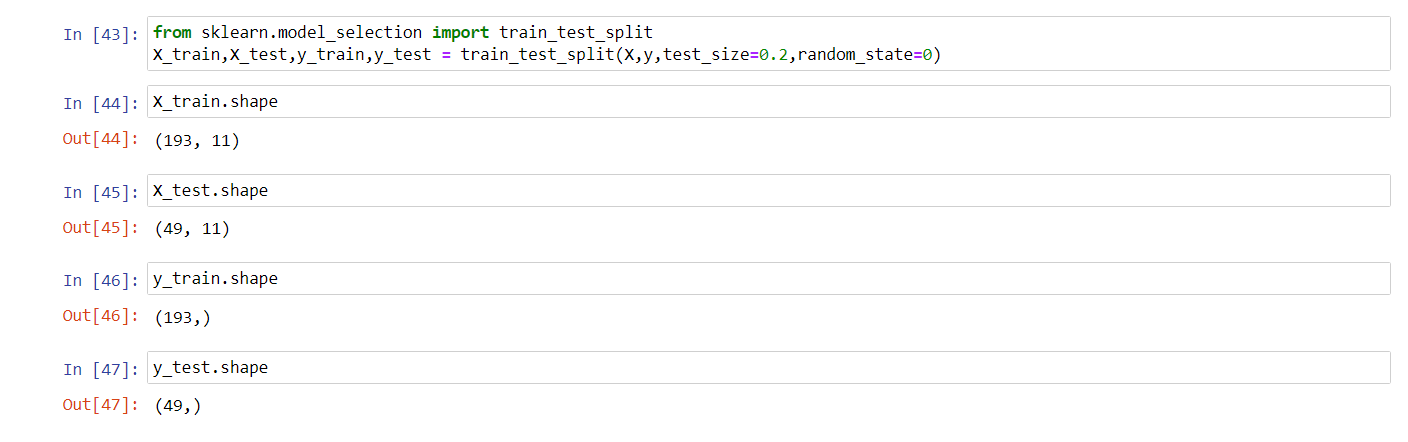
Here we’re dropping the unnecessary columns other than the ones with strong relationships and splitting the dataset into dependent variable y and independent variable X



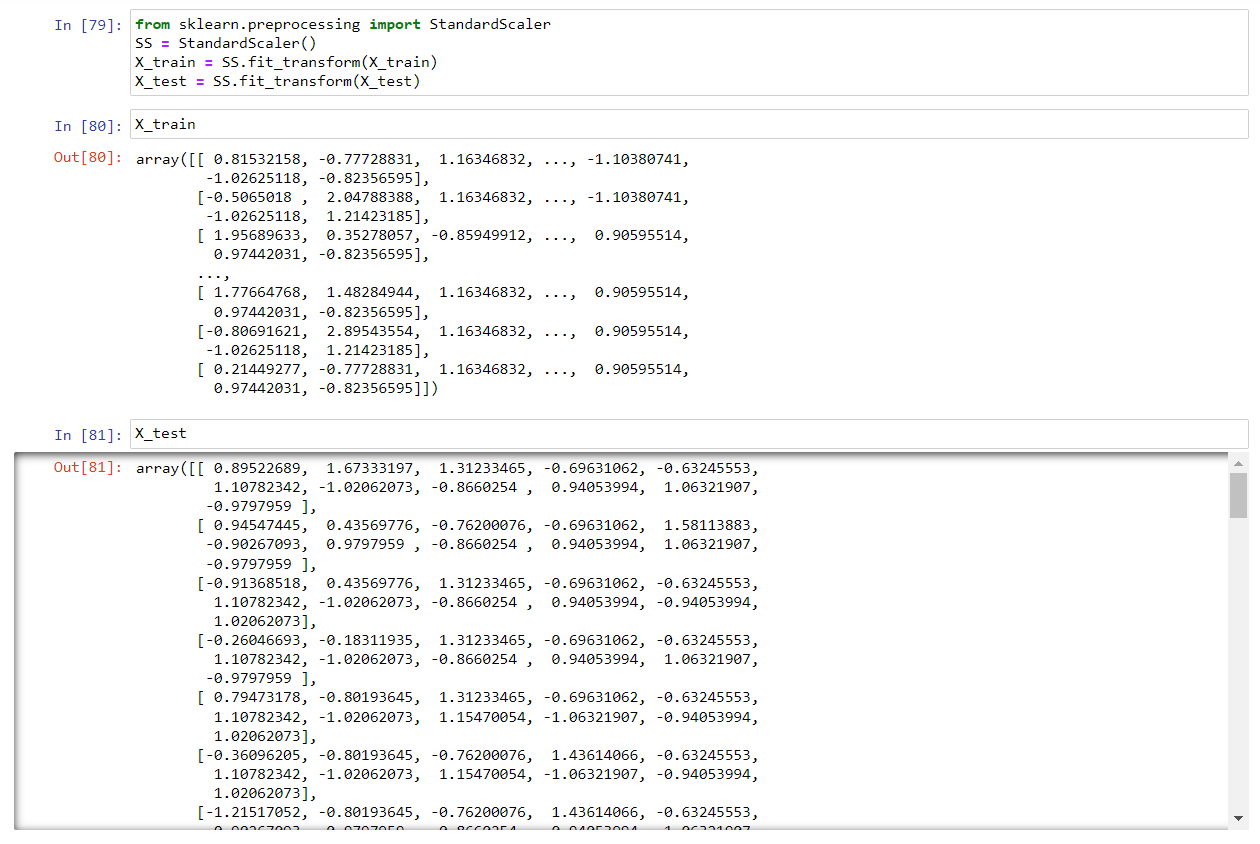
Here we’ve applied SMOTE oversampling technique to solve the imbalancement data.



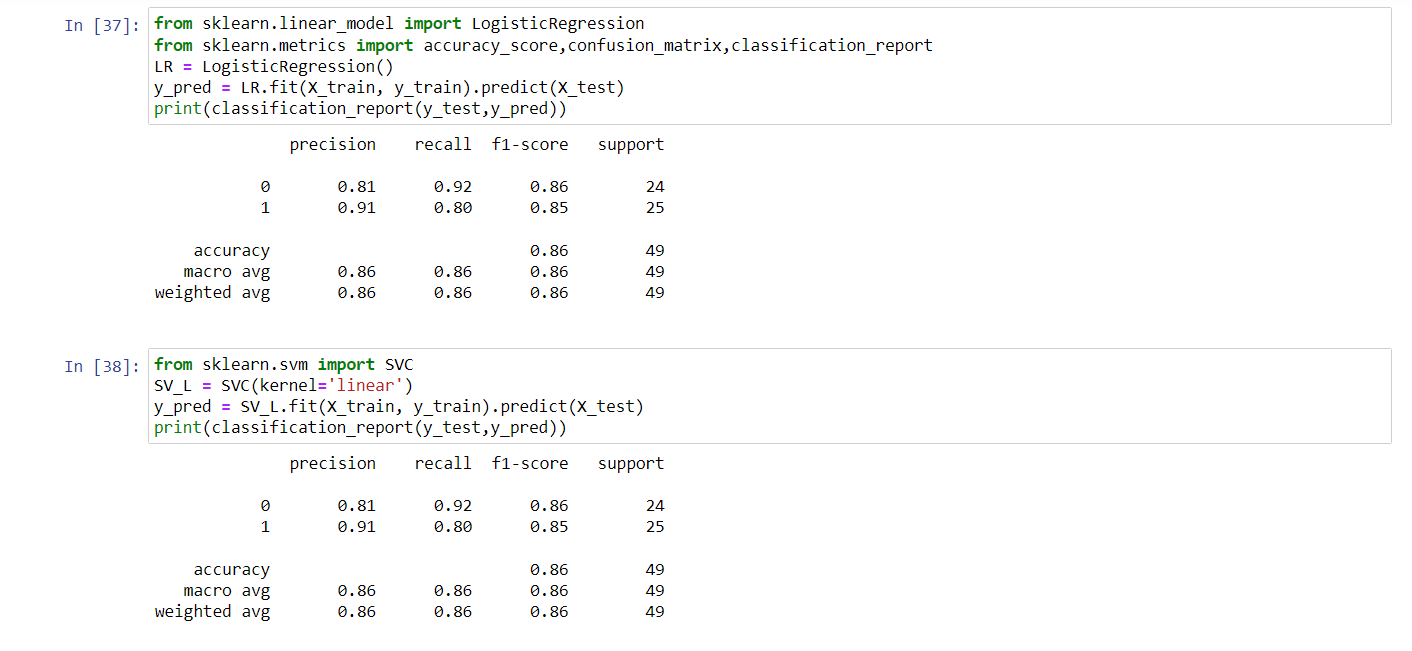
Here we’ve applied Label encoding technique to the dependent variable.



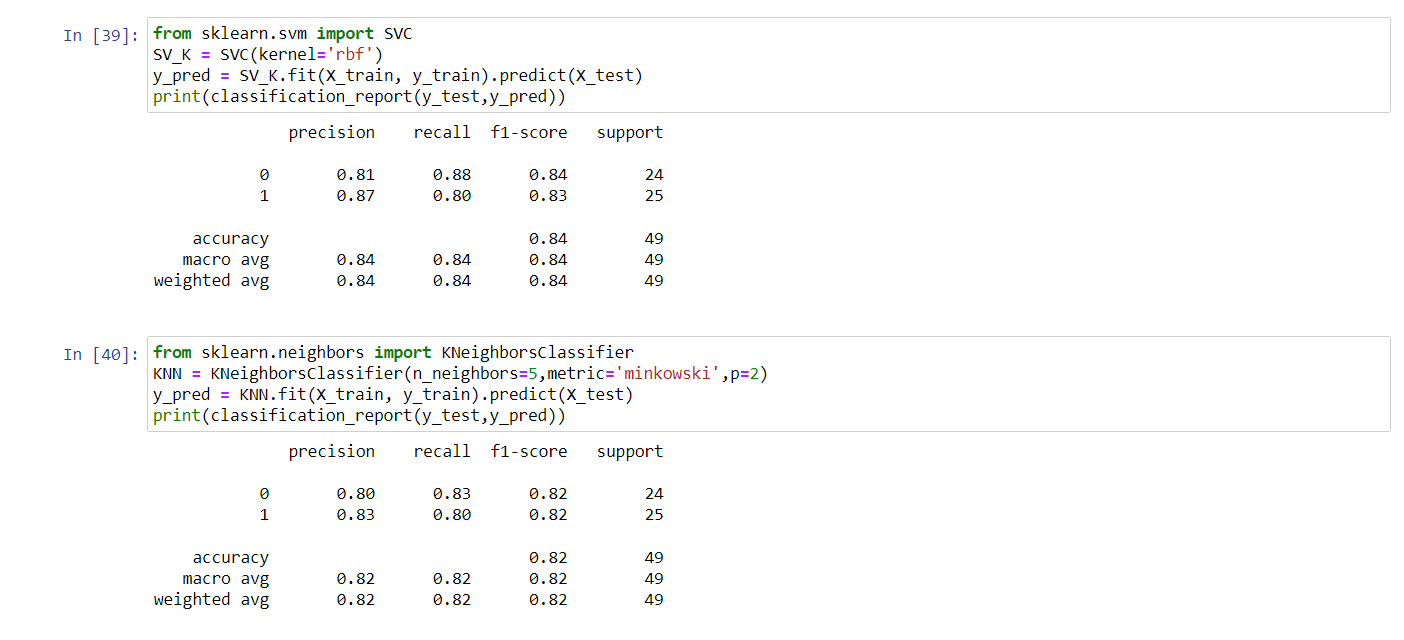
Here we’re separating the X and y into training and testing data with 80% for training and 20% for testing and using the train\_test\_split.



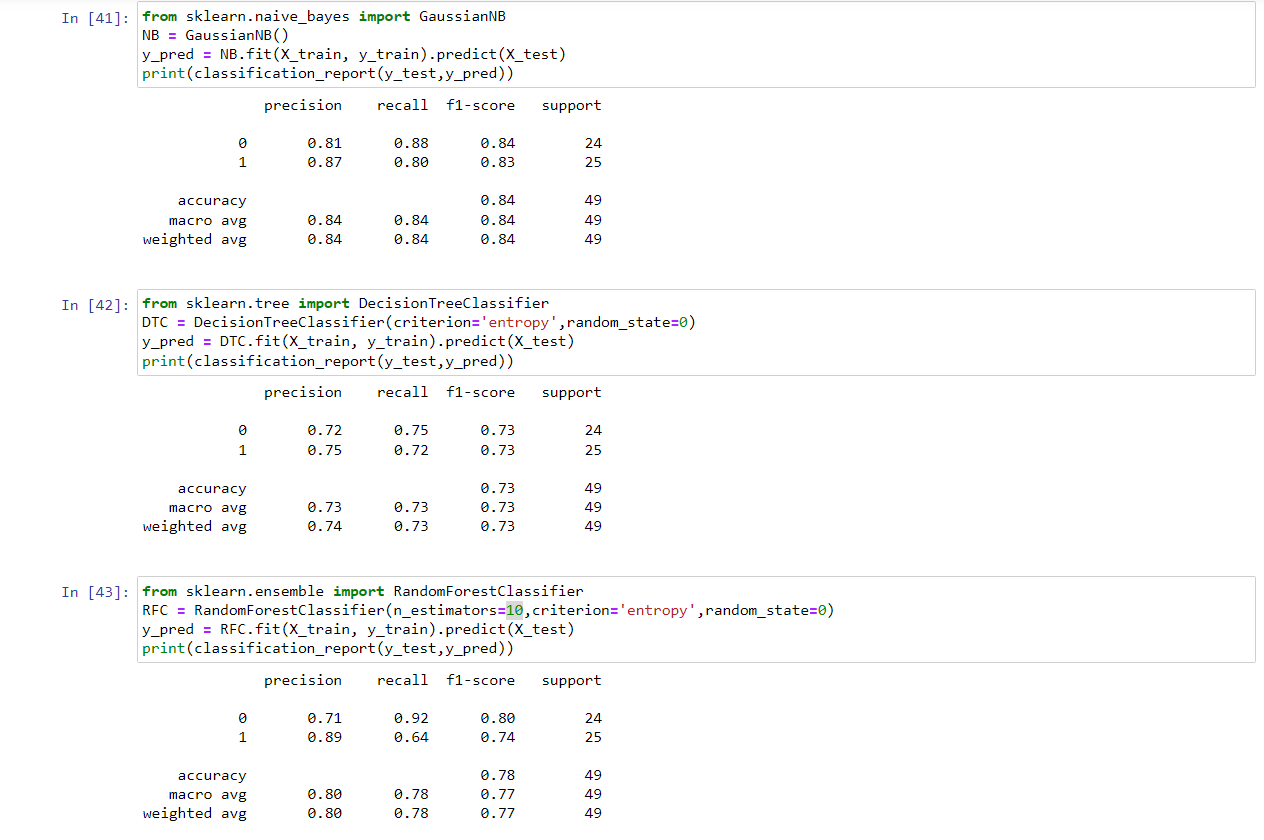
Here we’re applying Standard Scaling technique to the training data X\_test and X\_train to scale the independent variables



Applying Logistic Regression and Linear SVM



Applying Kernel SVM and KNN



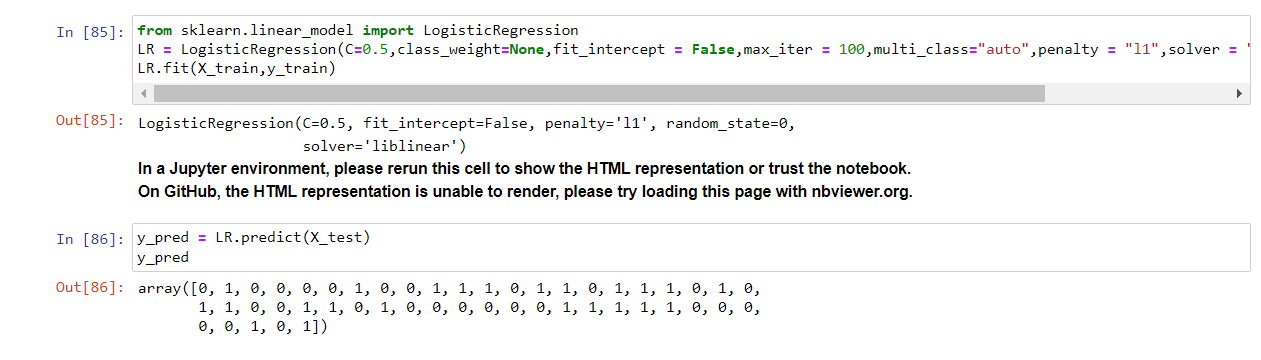
Applying Naive Bayes, Decision tree and Random Forest and we have found we got same accuracy in Logistic Regression and Linear SVM so let’s do Hyper parameter tuning for both



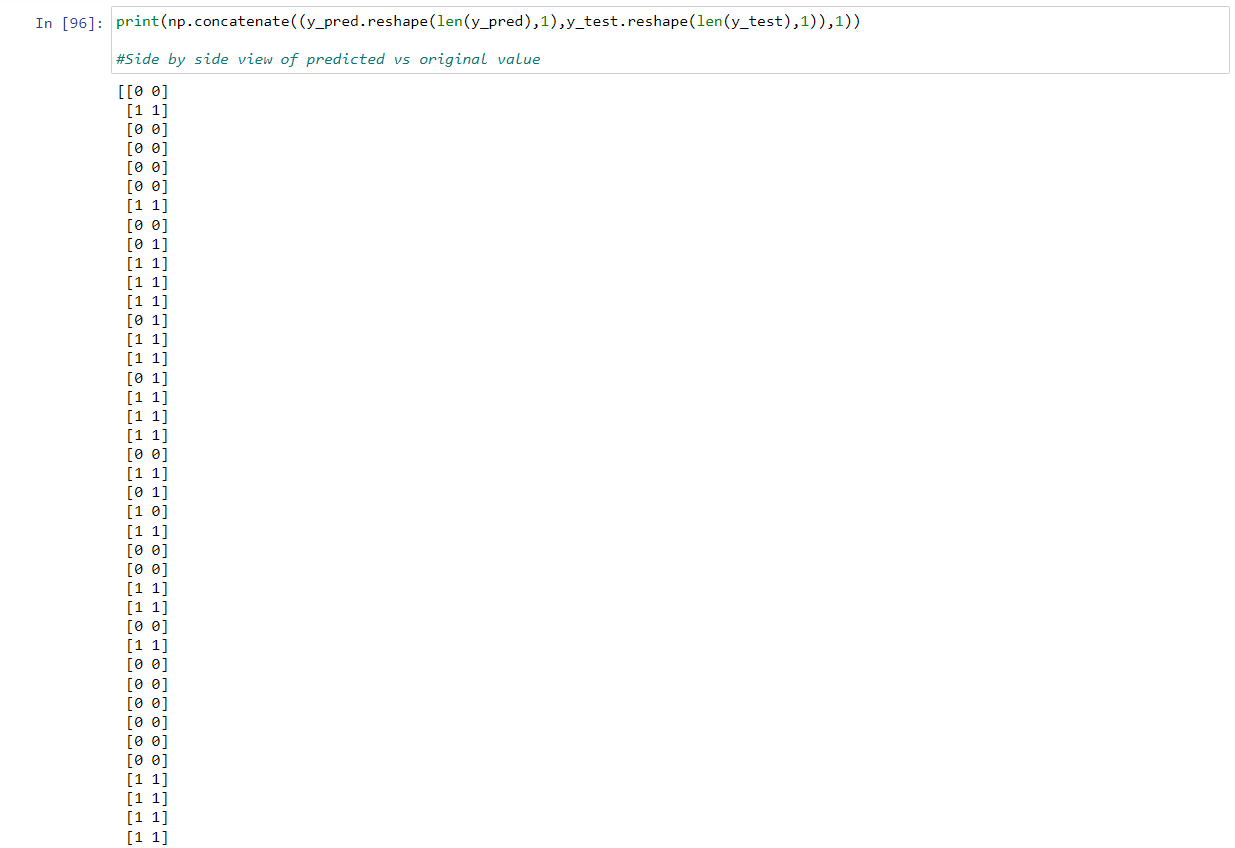
Here we’re applying GridSearchCV to the Logistic Regression model to get the best parameters for the dataset and getting the best accuracy and printing the best parameters



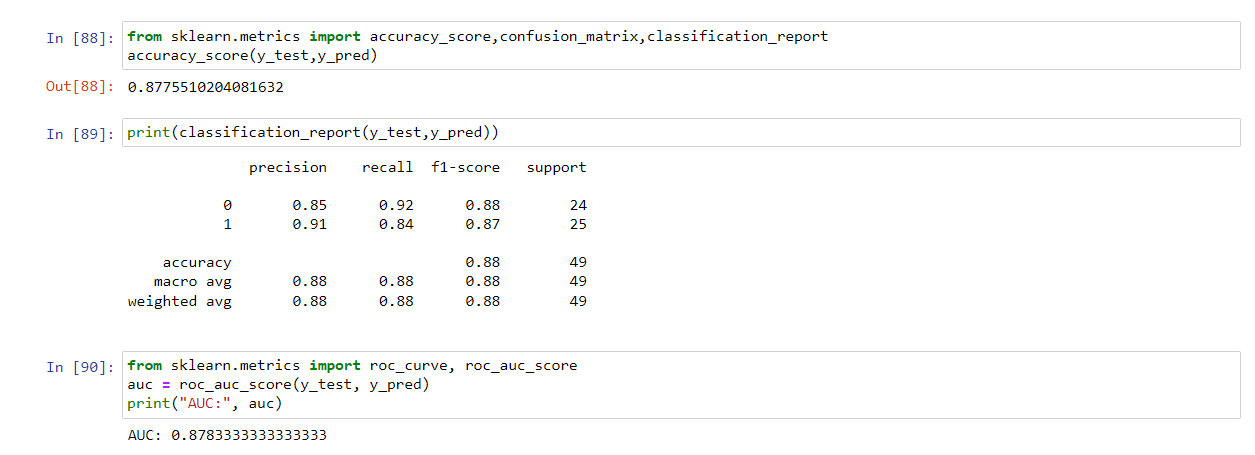
Here we’re applying GridSearchCV to the Linear SVM model to get the best parameters for the dataset and getting the best accuracy and printing the best parameters and we have high accuracy for Logistic Regression so let’s use that model



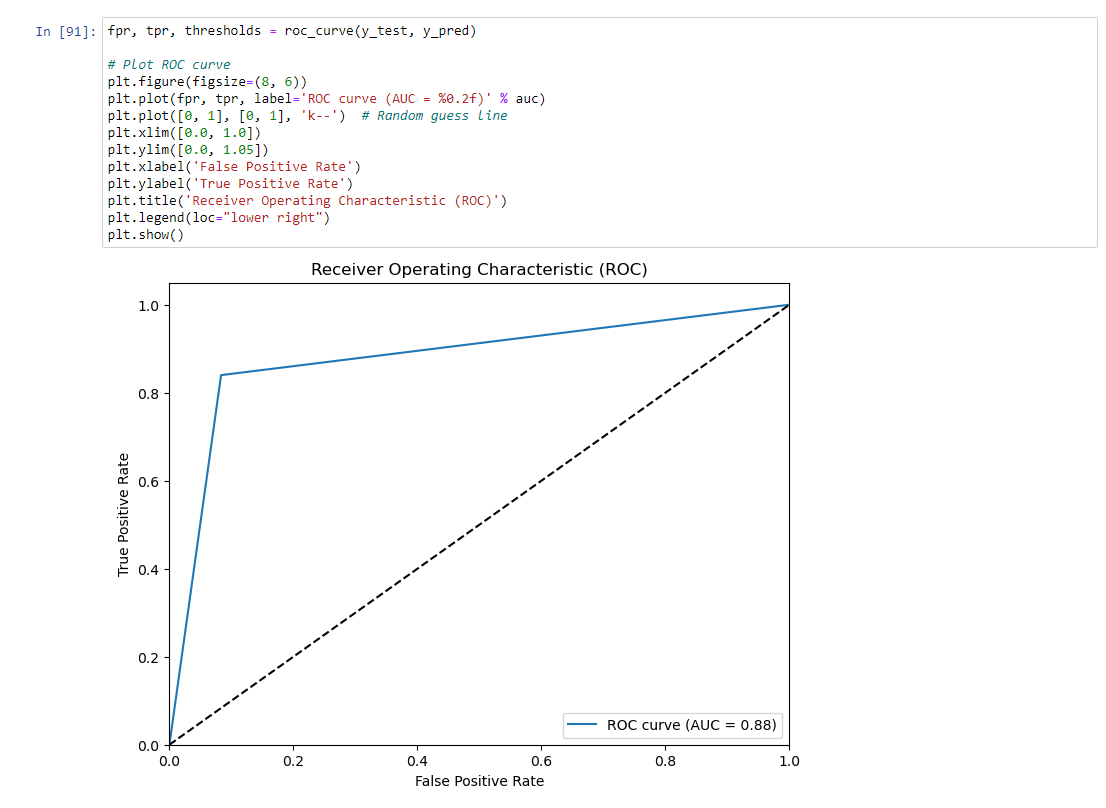
Here we’re fitting the model using Logistic Regression with the best parameters and predicting the X\_test’s target variable using predict and stored the result in y\_pred.



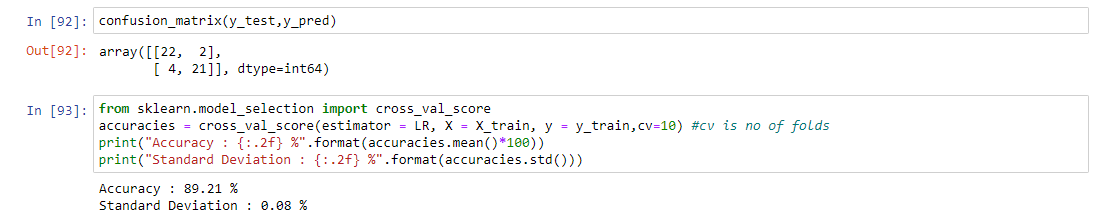
Here we’re just printing the predicted value side by side with the original value



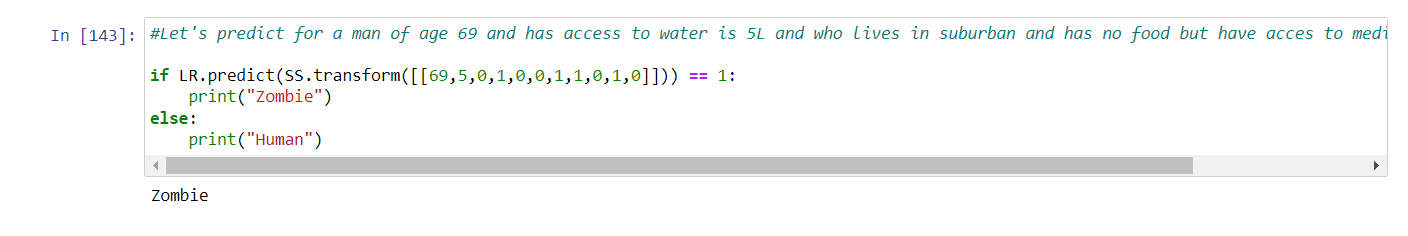
Here we’re printing the accuracy of the model and then printing the classification report and then printing the AUC score of it for drawing the ROC curve



This is the ROC curve for the model



Here we’re printing the confusion matrix for the model and printing the accuracy and standard deviation using the K-fold method with 10 folds using the cross\_val\_score.



Here the model predicted for a man of age 69 and has access to water is 5L and who lives in suburban and has no food but have access to medication and sanitation as Zombie



Here we’re creating the pickle format for model and scaling object as model.pkl and scaler.pkl to use it in the app.py for the Flask deployment

app.py

from flask import Flask, render\_template, request

import pickle

app = Flask(\_\_name\_\_)

model = pickle.load(open(r'C:/Users/Kapil/Desktop/Zombie Flask/model.pkl', 'rb'))

SS = pickle.load(open(r'C:/Users/Kapil/Desktop/Zombie Flask/scaler.pkl', 'rb'))

@app.route('/')

def helloworld():

return render\_template("index.html")

@app.route('/login', methods=['POST'])

def login():

a = request.form["age"]

r = request.form["rurality"]

f = request.form["food"]

m = request.form["medicine"]

s = request.form["sanitation"]

w = request.form["water"]

r1, r2, r3 = 0, 0, 0 # Assign default values

if r == "Urban":

r1, r2, r3 = 0, 0, 1

elif r == "Sub-Urban":

r1, r2, r3 = 0, 1, 0

elif r == "Rural":

r1, r2, r3 = 1, 0, 0

f1, f2 = 0, 0 # Assign default values

if f == "Food":

f1, f2 = 1, 0

elif f == "No Food":

f1, f2 = 0, 1

m1, m2 = 0, 0 # Assign default values

if m == "Medicine":

m1, m2 = 1, 0

elif m == "No Medicine":

m1, m2 = 0, 1

s1, s2 = 0, 0 # Assign default values

if s == "Sanitation":

s1, s2 = 1, 0

elif s == "No Sanitation":

s1, s2 = 0, 1

t = [[int(a), int(w), int(r1), int(r2), int(r3), int(f1), int(f2), int(m1), int(m2), int(s1), int(s2)]]

scaled\_input\_data = SS.transform(t)

output = model.predict(scaled\_input\_data)

if output == 1:

res = "Zombie"

else:

res = "Human"

return render\_template("index.html", y="It's a " + res)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

index.html

<!DOCTYPE html>

<html>

<head>

<title>Zombie Detector</title>

<link rel="stylesheet" type="text/css" href="{{ url\_for('static', filename='styles.css') }}">

</head>

<body>

<h1>Zombie Detector</h1>

<form action="/login" method="post">

<p class="dark-grey-text">Enter the details about the human and their available supplies!!</p>

<p>Age</p>

<p><input type="number" name="age" min="1" max="100" /></p>

<p>Water supply(in L)</p>

<p><input type="number" name="water" /></p>

<p>Rurality</p>

<select name="rurality">

<option value="">Select an option</option>

<option value="Urban">Urban</option>

<option value="Sub-Urban">Sub-urban</option>

<option value="Rural">Rural</option>

</select>

<p>Food Supply</p>

<select name="food">

<option value="">Select an option</option>

<option value="Food">Food</option>

<option value="No Food">No Food</option>

</select>

<p>Medicine Supply</p>

<select name="medicine">

<option value="">Select an option</option>

<option value="Medicine">Medicine</option>

<option value="No Medicine">No Medicine</option>

</select>

<p>Sanitation Supply</p>

<select name="sanitation">

<option value="">Select an option</option>

<option value="Sanitation">Sanitation</option>

<option value="No Sanitation">No Sanitation</option>

</select>

<p><input type="submit" value="Submit" /></p>

</form>

<b class="white-text">{{ y }}</b>

</body>

</html>

styles.css

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 20px;

background-image: url("zombie.jpeg");

background-size: cover;

background-position: center;

}

h1 {

color: #333;

text-align: center;

}

form {

max-width: 400px;

margin: 0 auto;

background-color: #fff;

padding: 20px;

border: 1px solid #ccc;

border-radius: 4px;

}

form p {

margin: 10px 0;

}

form select,

form input[type="number"] {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 4px;

}

form input[type="number"] {

width: calc(50% - 5px);

}

form input[type="submit"] {

background-color: #4CAF50;

color: white;

cursor: pointer;

padding: 10px 20px;

border: none;

border-radius: 4px;

}

b {

display: block;

text-align: center;

margin-top: 20px;

font-weight: bold;

}

.dark-grey-text {

color: #666;

}

select option {

display: none;

}

select:focus option {

display: block;

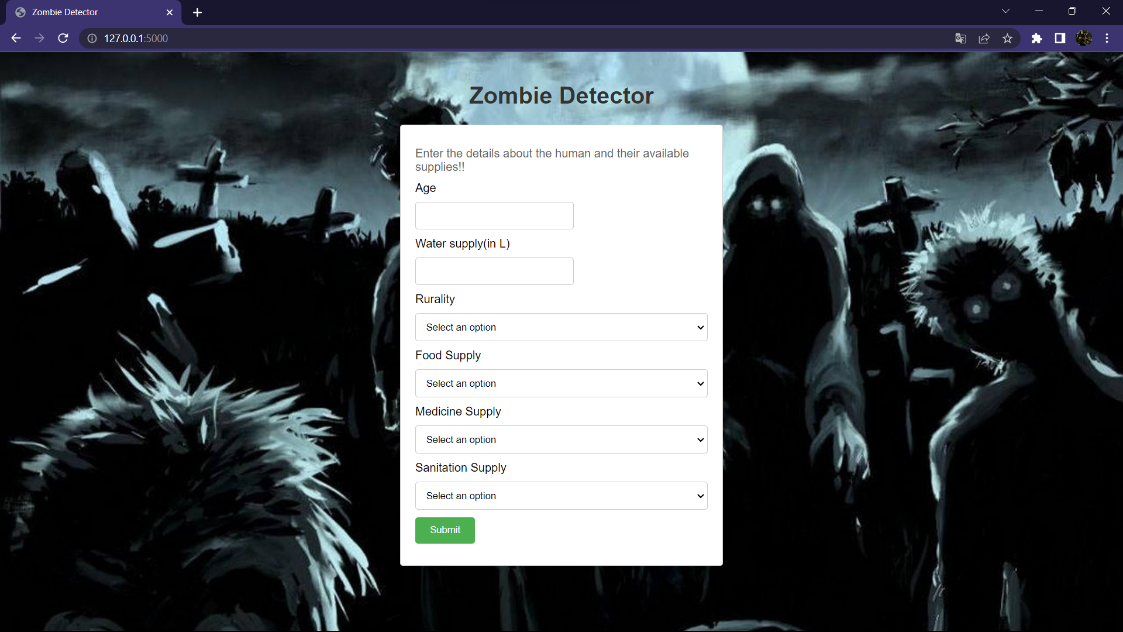
}

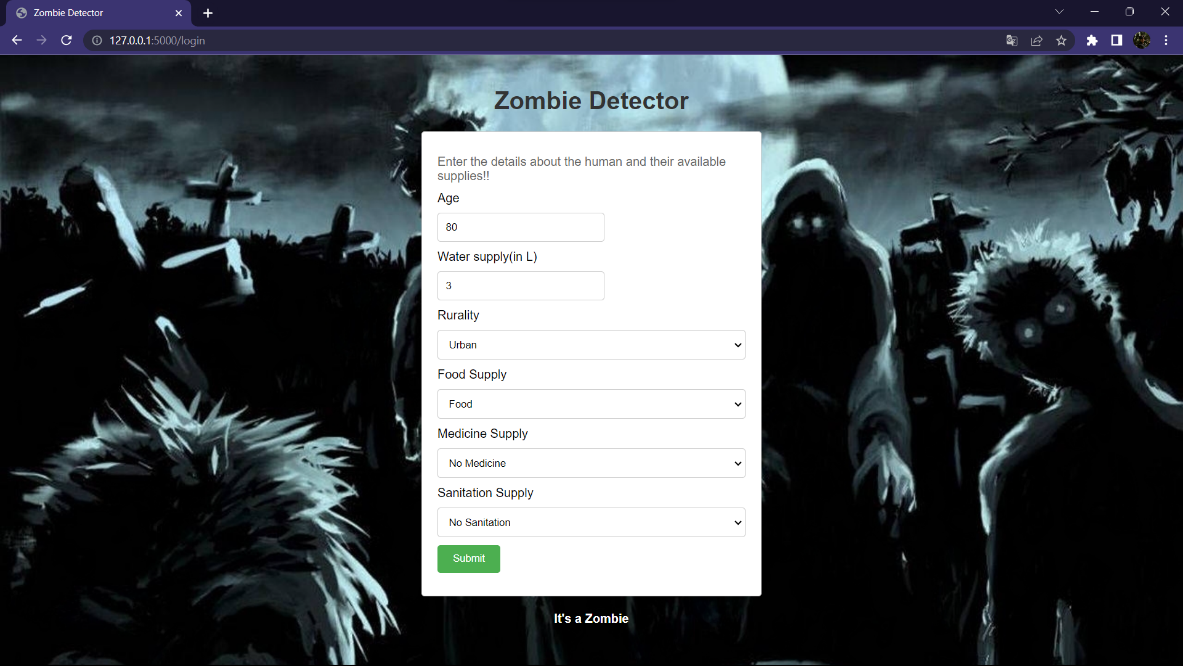
.white-text {

color: white;

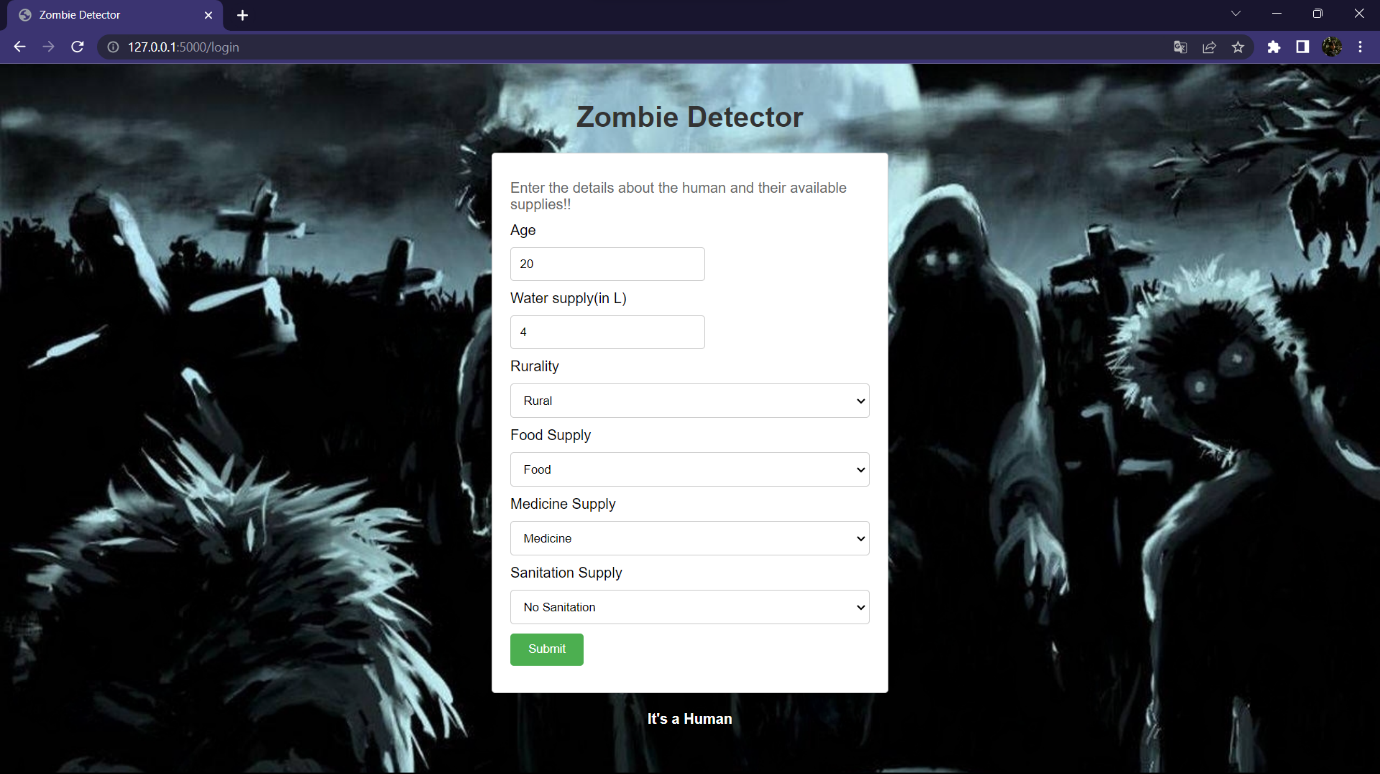
}

Output:



This is the GUI of the Flask deployment

This is the prediction for the Zombie



This is the prediction for the Human