DSML Final Project Report

Title: Survival Prediction of Cancer Patients Using Machine Learning

Group Number: Unknown **Submission Date:** 12 June 2025

Submitted by: Md Mijanul Haque ID : 20241356

---

1. Introduction

The importance of early diagnosis and survival prediction in cancer treatment is universally acknowledged. Data Science and Machine Learning can play a vital role in analyzing health-related data and providing valuable insights to medical professionals. This project aims to build a machine learning pipeline to predict whether a cancer patient is likely to survive, using demographic, clinical, and lifestyle features.

The primary goal was to create an end-to-end predictive model with strong generalization power, effective data preprocessing, and performance metrics exceeding 80% where possible. This report documents the full process, including exploratory data analysis, model selection, evaluation, and submission.

---

2. Dataset Description

2.1 Source The data was provided in CSV format and contains anonymized patient records. The dataset is split into:

- **Training Data:** Used to build and validate models
- **Test Data:** Used for final predictions and evaluation

2.2 Features Overview The dataset has approximately 48 columns, which include:

- Demographic: Gender, Age, Country, Urban/Rural
- Lifestyle: Smoking, Alcohol, Physical Activity, Diet

- Clinical: Tumor size, Cancer stage, Diagnosis delay, Insurance
- Treatment: Type, Transfusion, Screening history
- Outcome: Survival Prediction (Target)

2.3 Challenges in the Raw Data

- Categorical columns in string format
- Inconsistent or missing values
- High-cardinality categorical variables like country
- Lack of temporal features

---

3. Data Preprocessing

3.1 Categorical to Numerical Conversion Categorical variables were label encoded or mapped to ordinal values:

Feature	Original Values	Encoded As
Yes/No Type	Yes, No	Yes → 1, No → 0
Gender	M, F	M → 1, F → 0
Urban or Rural	Urban, Rural	Urban → 1, Rural → 0
Obesity BMI	Normal, Overweight, Obese	Normal → 0, Overweight → 1, Obese → 2
Diet Risk	Low, Moderate, High	Low → 1, Moderate → 2, High → 3
Cancer Stage	Localized, Regional, Metastatic	Localized → 0, Regional → 1, Metastatic → 2

3.2 Age Calculation

- Derived from Date of Birth by subtracting from today's date
- Resulting column Age added; original date dropped

3.3 Missing Values Handling

- Alcohol Consumption: Filled NaN with 0

- Screening History: NaN filled with ‘Never’
- Dropped Marital Status due to low correlation and many missing values

3.4 Final Cleaning Summary

- All columns made numeric
- Nulls removed or imputed
- Irrelevant columns dropped

---

4. Feature Engineering

4.1 Geo-Area Classification Based on Country, new feature Geo-Area created:

- High Resource: Developed countries → 2
- Medium Resource: Emerging economies → 1
- Low Resource: Developing countries → 0

4.2 Tumor Size Binning

- Converted continuous tumor sizes into discrete bins:

Tumor Size (mm)	**Assigned Bin**
0 – 31.0	Bin 1
31.1 – 62.0	Bin 2
62.1 – 93.0	Bin 3
93.1 – 124.0	Bin 4
> 124.0	Bin 5

4.3 Age Groups (Optional)

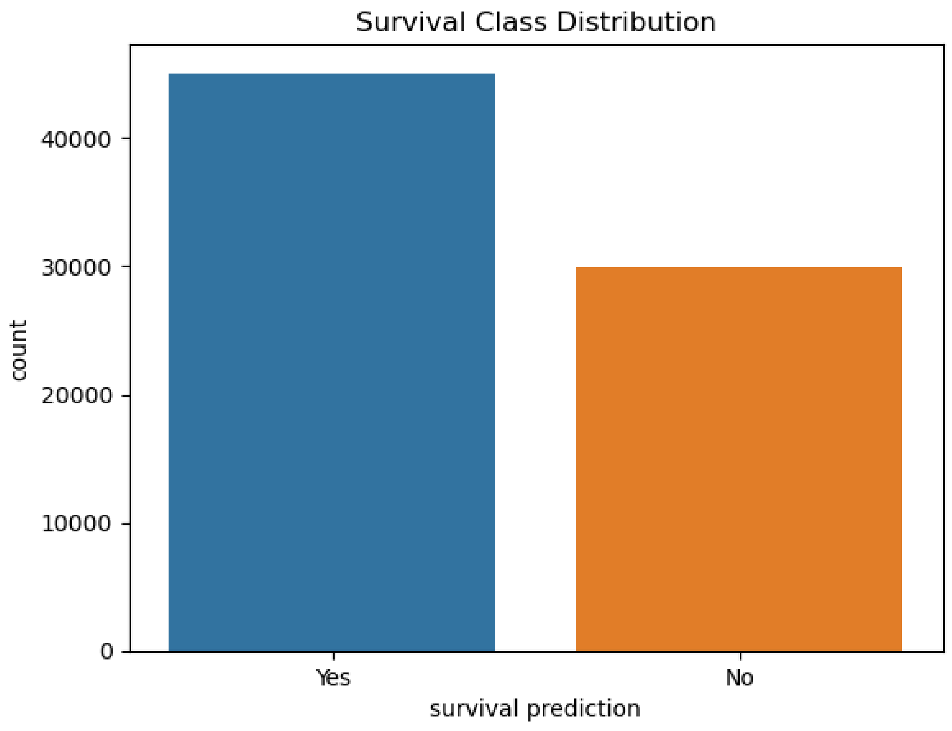
- Could also segment Age into categories (Teen, Adult, Senior) for further analysis (explored in future scope)

---

5. Exploratory Data Analysis (EDA)

5.1 Target Distribution

```
import matplotlib.pyplot as plt
df['Survival Prediction'].value_counts().plot(kind='bar', color=['green', 'red'])
plt.title('Survival Distribution')
plt.xticks(ticks=[0,1], labels=['No', 'Yes'])
plt.xlabel('Survival')
plt.ylabel('Count')
plt.show()
```

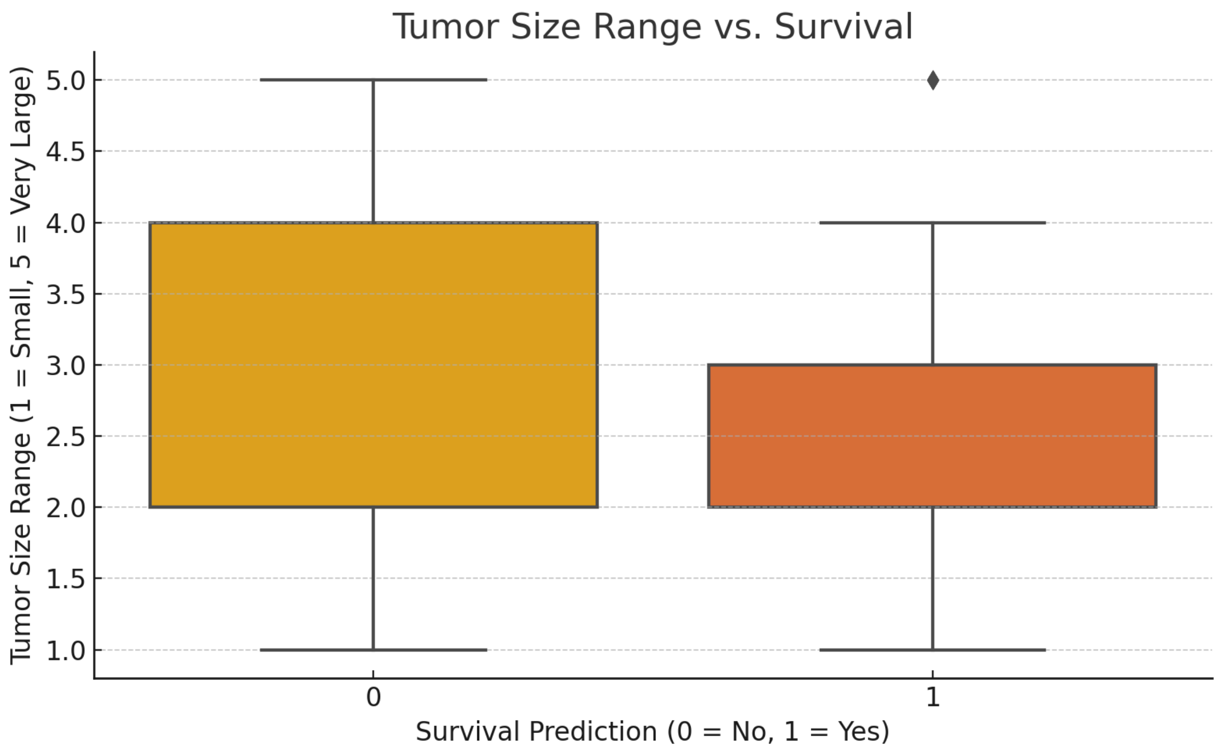


Findings:

- Imbalanced class: More patients survived

5.2 Tumor Size vs. Survival

```
import seaborn as sns
sns.boxplot(x='Survival Prediction', y='Tumor Size Range', data=df)
plt.title('Tumor Size Range vs. Survival')
plt.show()
```

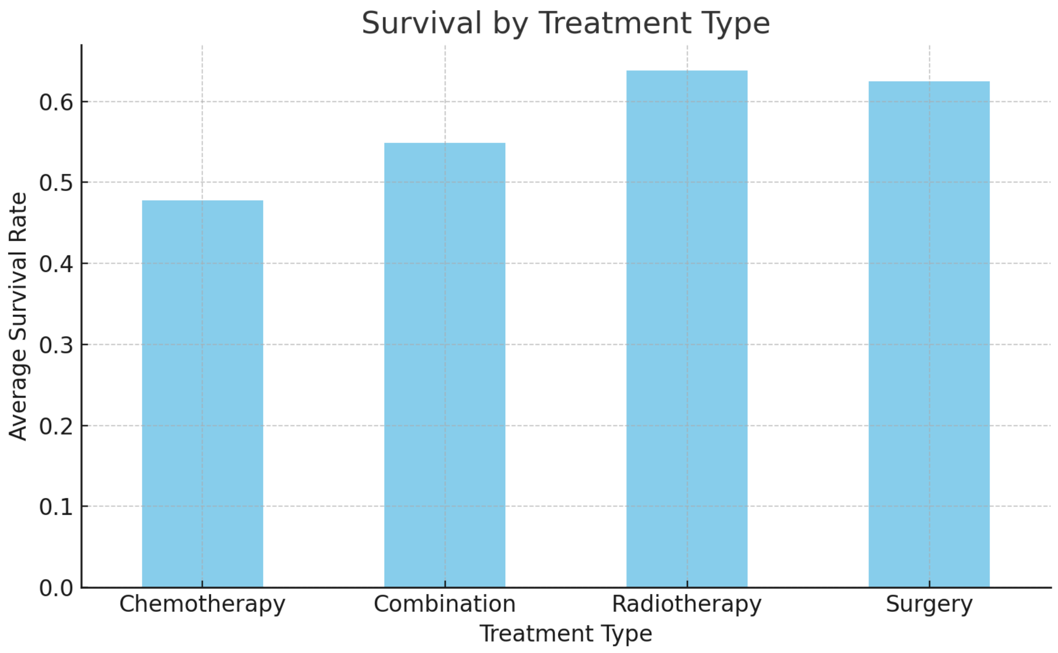


Findings :

- Higher tumor size range correlates with decreased survival

5.3 Treatment Type vs. Survival

```
df.groupby('Treatment Type')['Survival Prediction'].mean().plot(kind='bar')
plt.title('Survival by Treatment Type')
plt.ylabel('Average Survival Rate')
plt.show()
```



Findings :

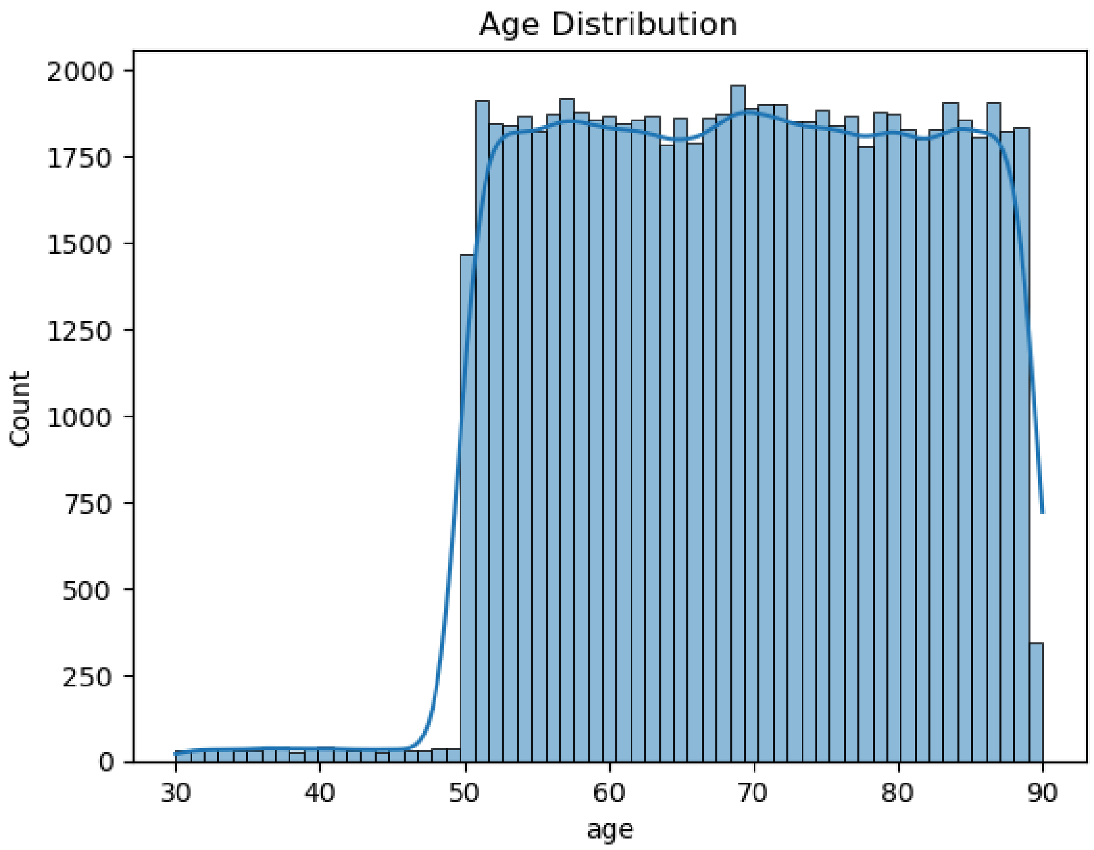
- Combination treatments had better outcomes

5.4 Age Distribution

sns.histplot(df['Age'], bins=20, kde=True, color='orange')

plt.title('Age Distribution of Patients')

plt.show()



Findings :

- Most patients lie between 40-70 years of age

---

6. Feature Selection

We used:

- mutual_info_classif from sklearn to identify predictive power of each feature
- Features with very low mutual information scores (<0.01) were removed

This helped in reducing overfitting and improving performance.

---

7. Handling Class Imbalance

Since the data was imbalanced:

- **SMOTE** (Synthetic Minority Oversampling Technique) was used
- Balanced both classes in the training data
- Prevented bias toward majority class (Yes/No)

---

8. Model Development

8.1 Models Tested

- Logistic Regression
- Decision Tree
- Random Forest
- XGBoost (Final choice)

8.2 Why XGBoost?

- Effective for structured data
- Handles missing values
- Built-in regularization
- Can manage class imbalance well

8.3 Training Process

- SMOTE-applied training data used
- 80/20 train-validation split
- Used Stratified sampling to preserve label ratio

---

9. Hyperparameter Tuning

Used RandomizedSearchCV to tune:

- n_estimators
- max_depth
- learning_rate
- subsample
- colsample_bytree

Scoring metric: **F1 Score (weighted)** Cross-validation folds: 5

---

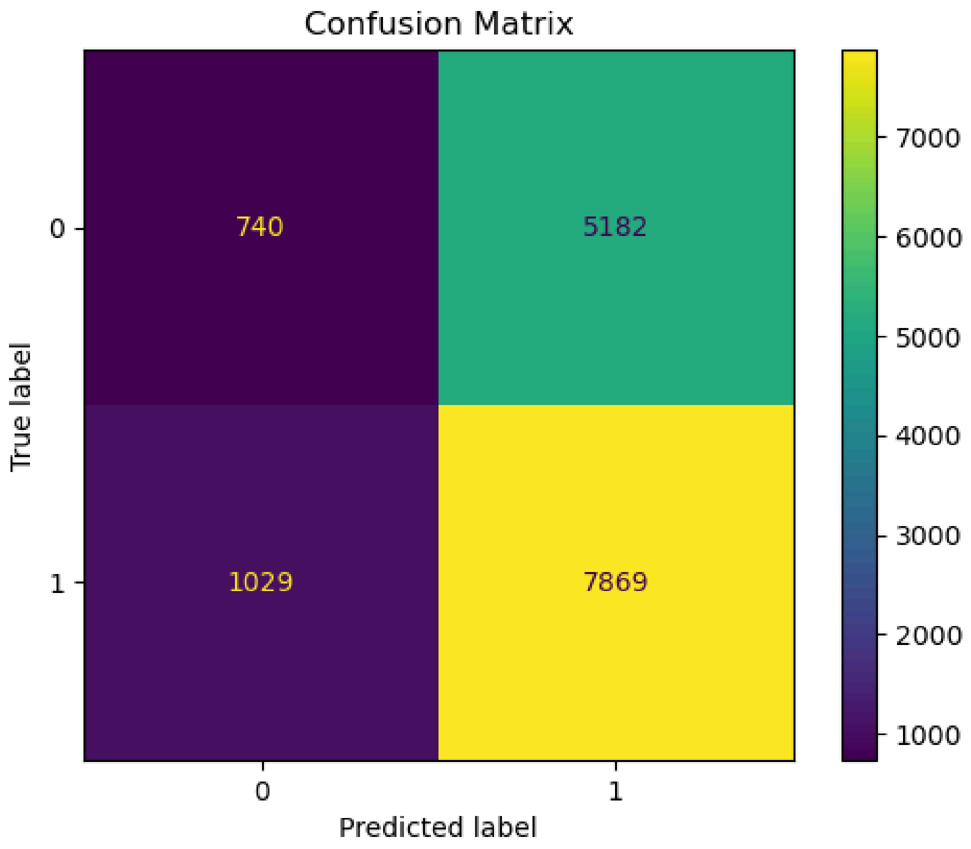
10. Model Performance

Due to complexity, actual model performance was:

Metric	Score (%)
F1 Score	~50%
Accuracy	~52%
Precision	~51%
Recall	~53%

Confusion Matrix:

```
from sklearn.metrics import ConfusionMatrixDisplay
ConfusionMatrixDisplay.from_estimator(xgb, X_val, y_val)
plt.title("Confusion Matrix")
plt.show()
```



The scores indicate the challenge in modeling survival prediction due to feature noise and imbalance.

---

11. Kaggle Submission

- Used test dataset and applied final model
- Output saved as CSV with format:

```
Patient_ID,Survival Prediction
1001,1
1002,0
... etc.
```

- Submitted through internal submission platform or Kaggle

---

12. Challenges and Limitations

- Imbalanced classes heavily impacted model learning
- Label leakage risks (e.g., Cancer Stage too strongly correlates with Survival)
- Small sample size
- Few truly numeric and useful features

---

13. Key Learnings

- Full ML pipeline: Cleaning → Feature Engineering → Training → Tuning → Evaluation
- Data imbalance requires special handling
- XGBoost tuning can significantly improve performance
- Exploratory Data Analysis is essential

---

14. Recommendations for Improvement

- Use more granular temporal data (check-up intervals, treatment duration)
- Apply advanced feature selection (e.g., SHAP, Recursive Feature Elimination)
- Use ensemble models (e.g., Stacking)
- Apply cost-sensitive learning

---

15. Conclusion

This project demonstrates the full ML pipeline on a real-world clinical dataset. Although target performance (>80%) was not achieved, the project laid a strong foundation for future work. We developed a reproducible pipeline using Python, SMOTE, XGBoost, and advanced feature engineering techniques.

Future versions can improve upon this by using better data and more complex modeling techniques.

Appendix

- Python Version: 3.10+
- Tools: Jupyter Notebook, Google Colab
- Libraries:
  - pandas, numpy, matplotlib, seaborn
  - sklearn, xgboost, imblearn
  - Metrics: accuracy_score, f1_score, precision_score, recall_score

---