HTETOPS

Final Project

Predictive Modeling of Cancer Patient Outcomes

Survival Duration, Mortality, and Cause of Death Classification

Presented by

Htet Aung Phyo
Htet Myet Zaw
Htet Kay Khine

Date

30 August 2025

Mentor

Ma Aye Khin Khin Hpone

TABLE OF CONTENT

HTETOPS

The Problem We Solve

Cancer is still a leading cause of death worldwide. Early prediction can save lives, helping doctors choose better treatments, reduce unnecessary procedures, and improve patients' quality of life. With powerful machine learning and SEER data, this project turns raw medical records into actionable insights for better survival outcomes.

01.

Survival Duration Classification

Classify patients into short-term (0-36 months) or long-term (36+ months) survival categories after diagnosis.

02.

Vital Status Prediction

Predict whether the patient is alive or dead and provide an estimated mortality probability.

03.

Cause of Death Prediction

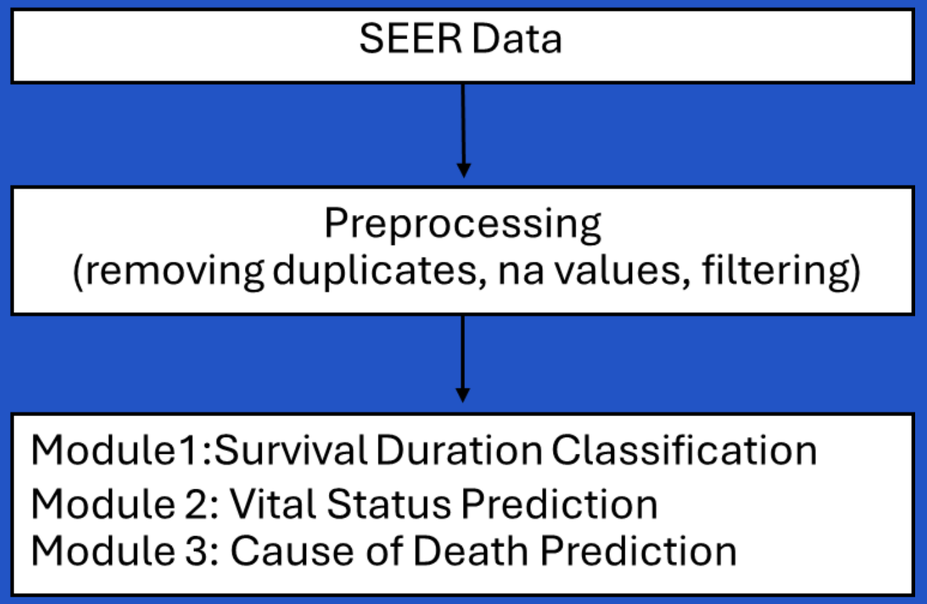
Determine if a patient died from cancer or another cause while suffering from cancer.

SEER DATASET

Source **National Cancer Institute (US)**

Data Records **1.4 M**

We will use the SEER Research Data (Incidence – SEER 8 Registries, Nov 2024 Submission), which contains patient-level cancer incidence and survival information from 2004 to 2015.



Demographics

Age, sex, race, year of diagnosis.

Socioeconomic

Household income, rural-urban classification.

Tumour

Site, morphology, stage

Treatment

Surgery, chemotherapy, radiation therapy, and treatment sequence.

MODULE 1

Survival Duration classification

Model Selection

- Logistics Regression
- Support Vector Machine

Library Used

- sklearn.model_selection
- sklearn.preprocessing
- sklearn.pipeline
- sklearn.linear_model
- matplotlib.pyplot
- numpy / pandas

Commands + Code + Text Run all

```
mat_clf = confusion_matrix(y_test, ypred_test)
report_clf = classification_report(y_test, ypred_test)

print("**Logistic Regresion**")
print(mat_clf)
print(report_clf)

ypred_testP = LR_pipeline.predict_proba(X_test)
auc = roc_auc_score(y_test, ypred_testP[:,1])
print('Accuracy:' ,auc)
```

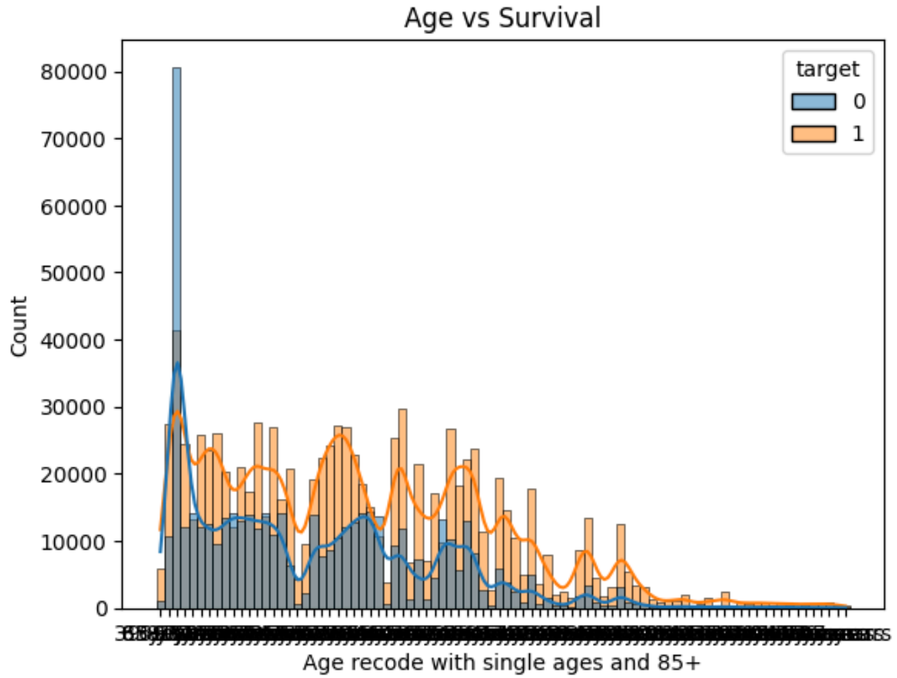
```
**Logistic Regresion**
[[ 40025  12881]
 [ 44099 145577]]
              precision    recall  f1-score   support

           0       0.48      0.76      0.58     52906
           1       0.92      0.77      0.84    189676

    accuracy                           0.77    242582
   macro avg       0.70      0.76      0.71    242582
weighted avg       0.82      0.77      0.78    242582

Accuracy: 0.8394584754981451
```

Commands + Code + Text Run all

```
ypred_train = LR_pipeline.predict(X_train)
mat_clf = confusion_matrix(y_train, ypred_train)
report_clf = classification_report(y_train, ypred_train)

print(mat_clf)
print(report_clf)

ypred_trainP = LR_pipeline.predict_proba(X_train)
auc = roc_auc_score(y_train, ypred_trainP[:,1])
print('Accuracy:' ,auc)
```

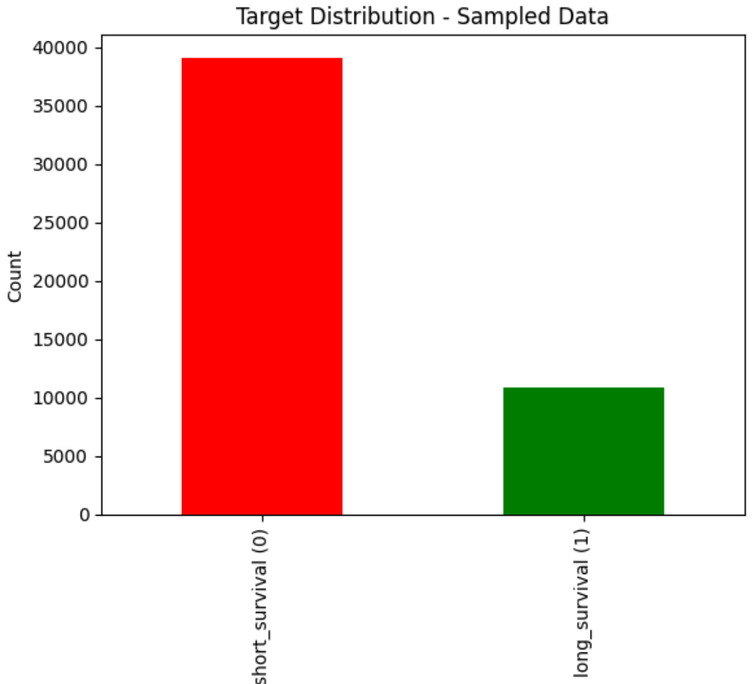
```
[[ 60068  18943]
 [ 66705 218156]]
              precision    recall  f1-score   support

           0       0.47      0.76      0.58     79011
           1       0.92      0.77      0.84    284861

    accuracy                           0.76    363872
   macro avg       0.70      0.76      0.71    363872
weighted avg       0.82      0.76      0.78    363872

Accuracy: 0.8408337469781741
```

Variables Terminal



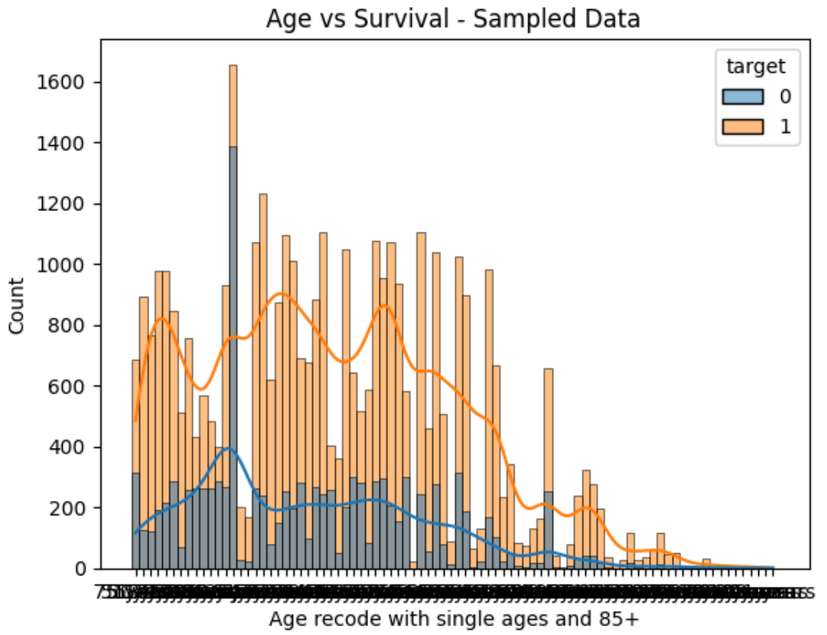
```python
#------------------------------
ypred_test = svm_pipeline.predict(X_test)
print("**Linear SVM - Test Set**")
print(confusion_matrix(y_test, ypred_test))
print(classification_report(y_test, ypred_test))

# ROC-AUC using decision function
ypred_test_scores = svm_pipeline.decision_function(X_test)
auc = roc_auc_score(y_test, ypred_test_scores)
print('ROC-AUC:', auc)
```

```
**Linear SVM - Test Set**
[[ 3309  1083]
 [ 3706 11902]]
              precision    recall  f1-score   support

           0       0.47      0.75      0.58      4392
           1       0.92      0.76      0.83     15608

    accuracy                           0.76     20000
   macro avg       0.69      0.76      0.71     20000
weighted avg       0.82      0.76      0.78     20000

ROC-AUC: 0.8354342201911307
```



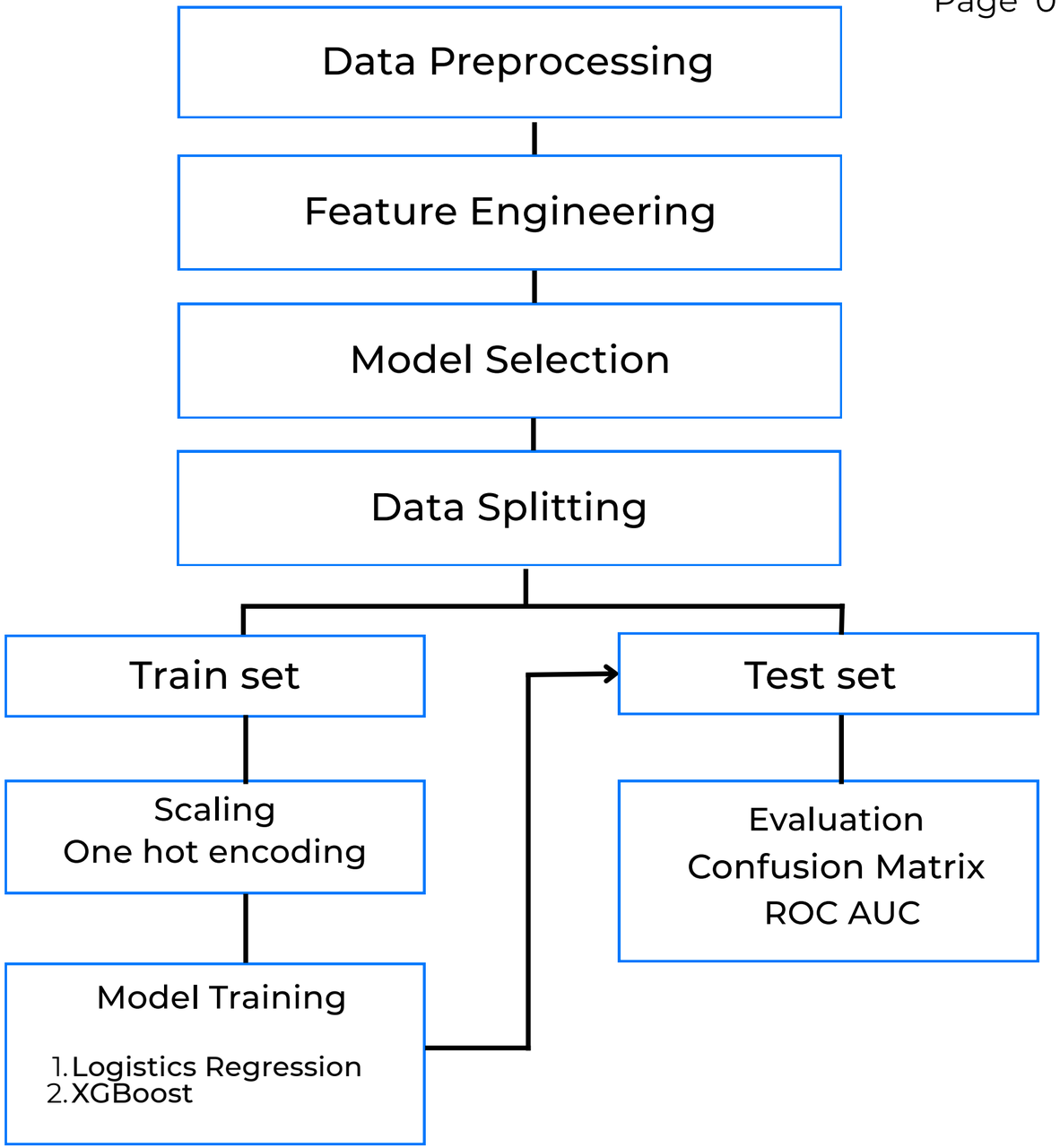

```python
#------------------------------
ypred_train = svm_pipeline.predict(X_train)
print("**Linear SVM - Train Set**")
print(confusion_matrix(y_train, ypred_train))
print(classification_report(y_train, ypred_train))

ypred_train_scores = svm_pipeline.decision_function(X_train)
auc_train = roc_auc_score(y_train, ypred_train_scores)
print('Train ROC-AUC:', auc_train)
```

```
**Linear SVM - Train Set**
[[ 4970  1528]
 [ 5308 18194]]
              precision    recall  f1-score   support

           0       0.48      0.76      0.59      6498
           1       0.92      0.77      0.84     23502

    accuracy                           0.77     30000
   macro avg       0.70      0.77      0.72     30000
weighted avg       0.83      0.77      0.79     30000

Train ROC-AUC: 0.8476413433469013
```



MODULE 2

Mortality Risk Prediction



```
Index(['age', 'sex', 'race', 'marital', 'primary_site', 'histology', 'grade',
       'tumor_size', 'extension', 'lymph_nodes', 'mets', 'ajcc_stage',
       'vital_status'],
      dtype='object')
```

Dataset shape: (1278767, 13)

Training set: (1023013, 470)
Test set: (255754, 470)

Target distribution: {1: 717663, 0: 561104}

```
Training Logistic Regression...
Logistic Regression - ROC AUC: 0.8747
Training XGBoost...
XGBoost - ROC AUC: 0.8858
```

Mortality Risk Prediction

Evaluation on Train Set

```
=== TRAINING SET RESULTS ===
ROC AUC: 0.8860
Sensitivity: 0.807 (80.7% of deaths identified)
Specificity: 0.797 (79.7% of alive correctly identified)
Precision: 0.836
F1 Score: 0.821

Confusion Matrix (Training):
True Negatives: 357688, False Positives: 91195
False Negatives: 110522, True Positives: 463608

Classification Report (Training):
              precision    recall  f1-score   support

           0       0.76      0.80      0.78    448883
           1       0.84      0.81      0.82    574130

    accuracy                           0.80   1023013
   macro avg       0.80      0.80      0.80   1023013
weighted avg       0.80      0.80      0.80   1023013
```

Evaluation on Test Set

```
=== TEST SET RESULTS ===
ROC AUC: 0.8858
Sensitivity: 0.807 (80.7% of deaths identified)
Specificity: 0.797 (79.7% of alive correctly identified)
Precision: 0.836
F1 Score: 0.821

Confusion Matrix (Test):
True Negatives: 89445, False Positives: 22776
False Negatives: 27651, True Positives: 115882

Classification Report (Test):
              precision    recall  f1-score   support

           0       0.76      0.80      0.78    112221
           1       0.84      0.81      0.82    143533

    accuracy                           0.80    255754
   macro avg       0.80      0.80      0.80    255754
weighted avg       0.80      0.80      0.80    255754
```

Cross Validation

```
=== Cross-Validation Results for XGBoost ===
ROC_AUC:
  Mean: 0.8841 (+/- 0.0015)
  Individual folds: ['0.8839', '0.8830', '0.8850', '0.8849', '0.8837']
PRECISION:
  Mean: 0.8340 (+/- 0.0009)
  Individual folds: ['0.8336', '0.8338', '0.8349', '0.8340', '0.8339']
RECALL:
  Mean: 0.8065 (+/- 0.0019)
  Individual folds: ['0.8071', '0.8048', '0.8074', '0.8072', '0.8061']
F1:
  Mean: 0.8200 (+/- 0.0012)
  Individual folds: ['0.8201', '0.8191', '0.8209', '0.8204', '0.8197']
```

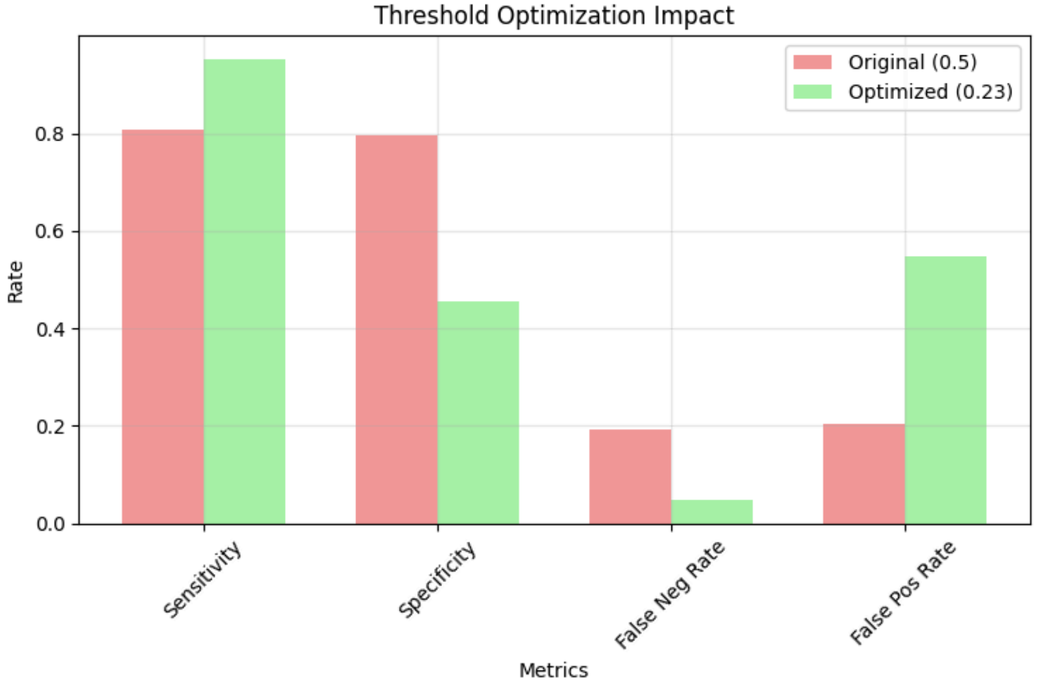
Default Threshold

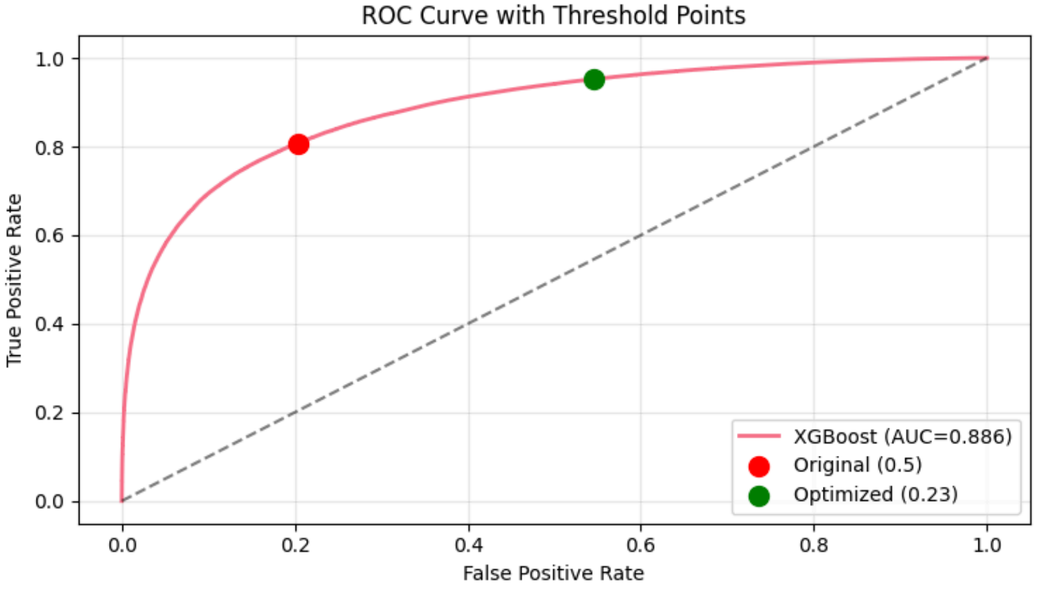
```
=== DETAILED EVALUATION OF XGBOOST ===
              precision    recall  f1-score   support

           0       0.76      0.80      0.78    112221
           1       0.84      0.81      0.82    143533

    accuracy                           0.80    255754
   macro avg       0.80      0.80      0.80    255754
weighted avg       0.80      0.80      0.80    255754


Clinical Metrics:
Sensitivity: 0.807 (80.7% of deaths identified)
Specificity: 0.797 (79.7% of alive correctly identified)
```

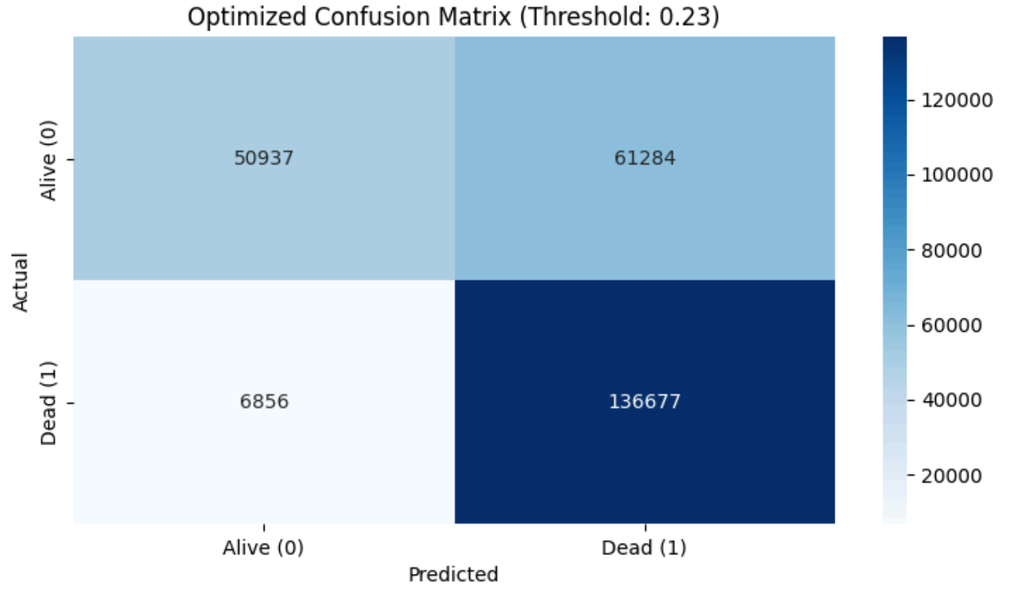
Optimized Threshold

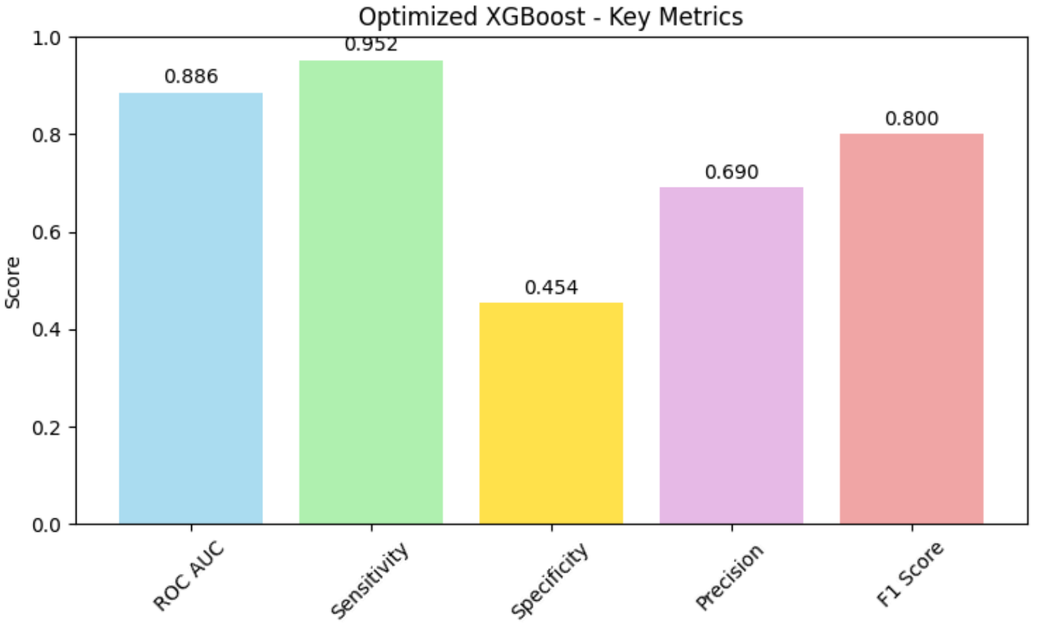
```
=== OPTIMAL THRESHOLD METRICS ===
         Metric    Value
      Threshold    0.230
    Sensitivity    0.952
    Specificity    0.454
      Precision    0.690
       F1 Score    0.800
False Positives 61,284.0
False Negatives  6,856.0
```

Cancer Prediction Model Analysis - Optimized Threshold



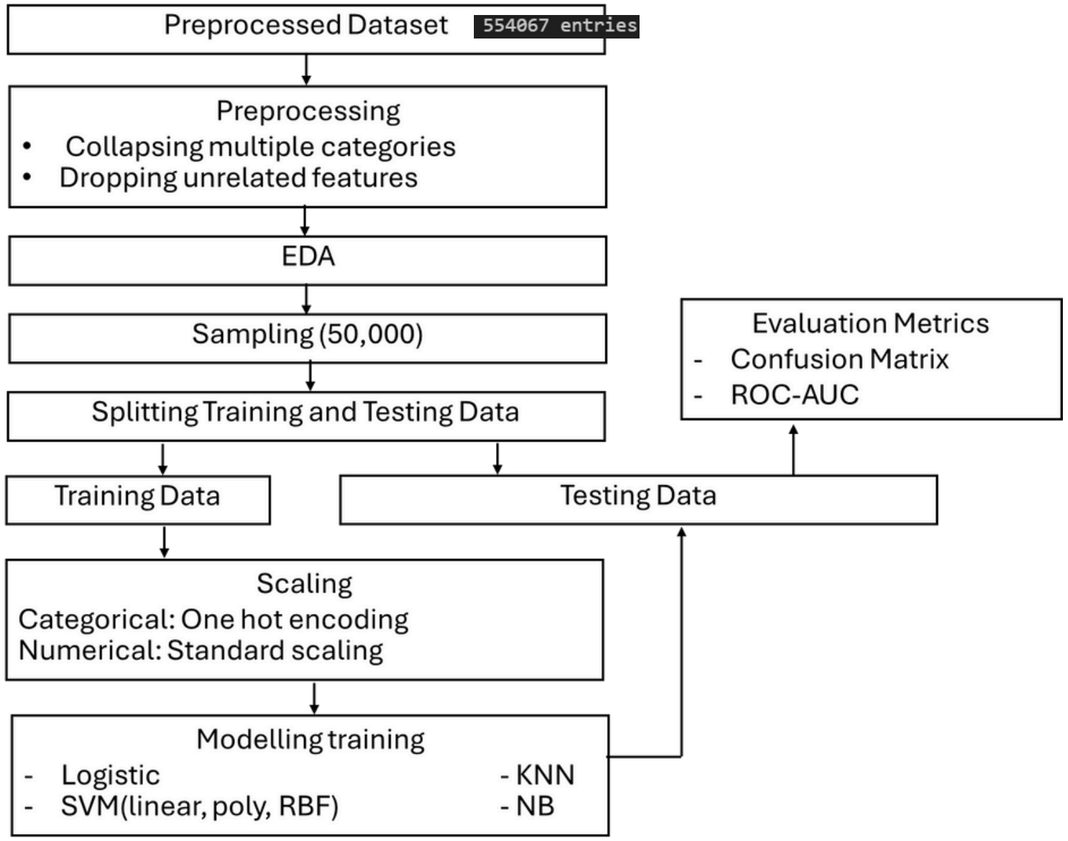




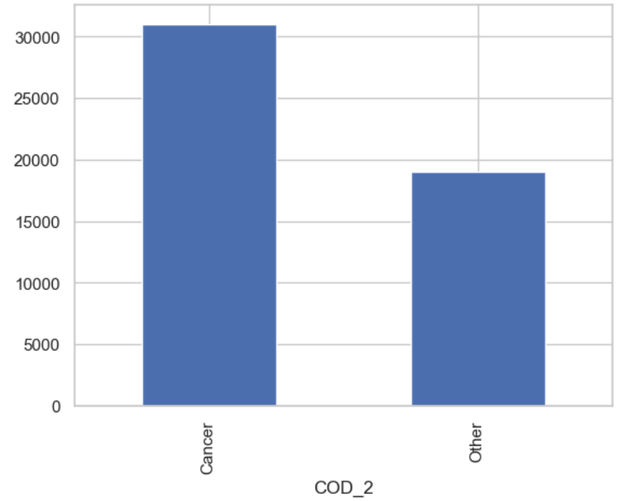


MODULE 3

Cause of Death Classification

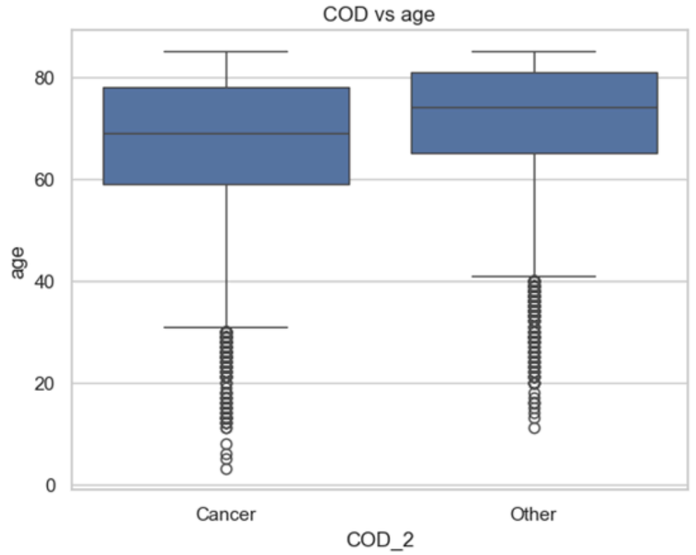


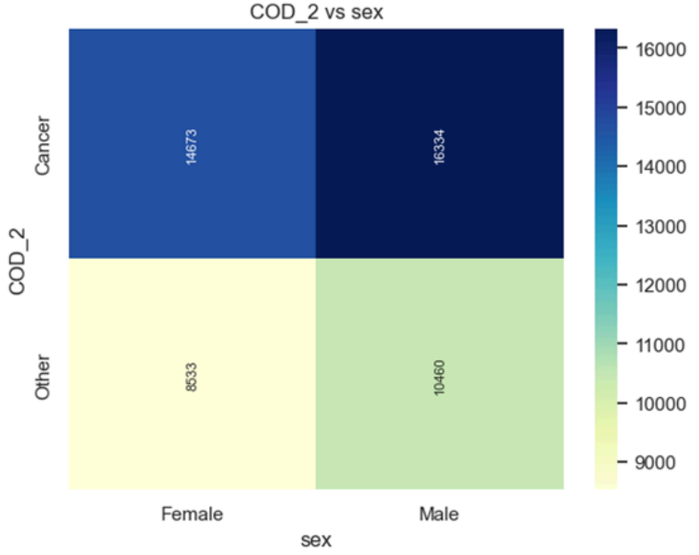

```
X.columns
✓ 0.0s
Index(['age', 'sex', 'race', 'marital_status', 'tumor_size', 'tumor_spread',
       'tumor_lymph_nodes', 'surgery_primary', 'chemotherapy',
       'PrimarySiteGroup', 'income_cat', 'location_collapsed',
       'AJCC_collapsed', 'radiation_collapsed'],
      dtype='object')
```

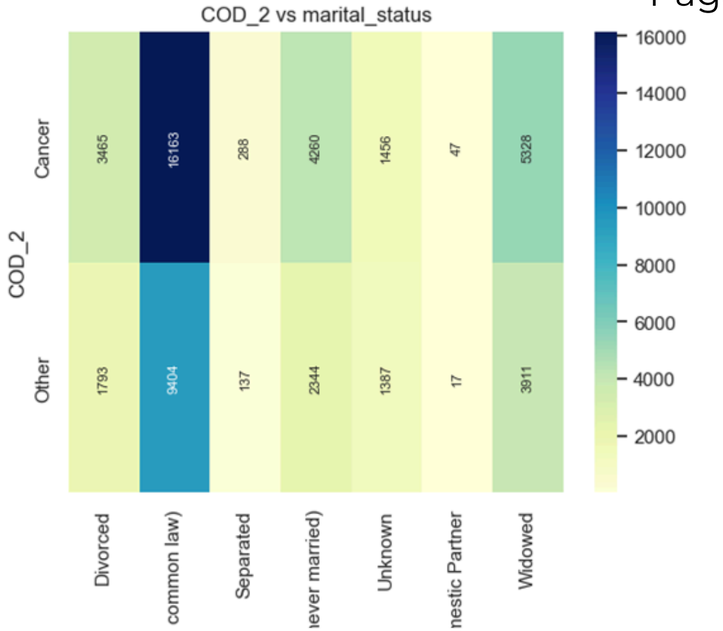


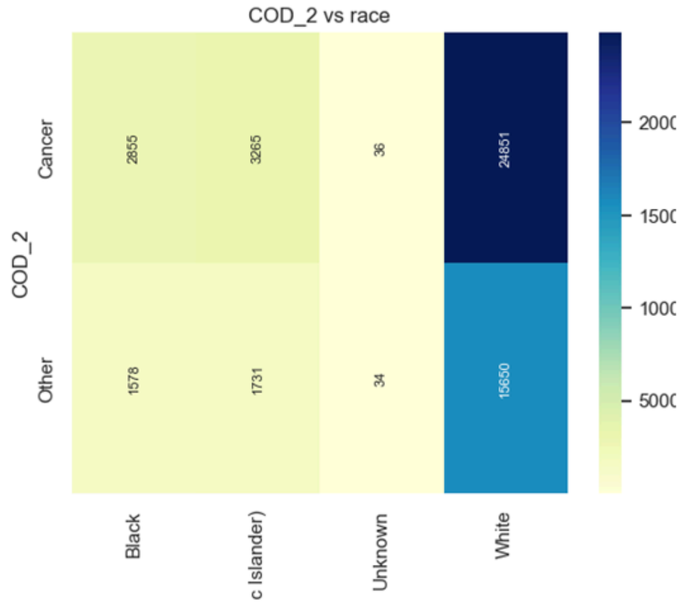
library used; pandas, numpy, seaborn, matplotlib,sklearn

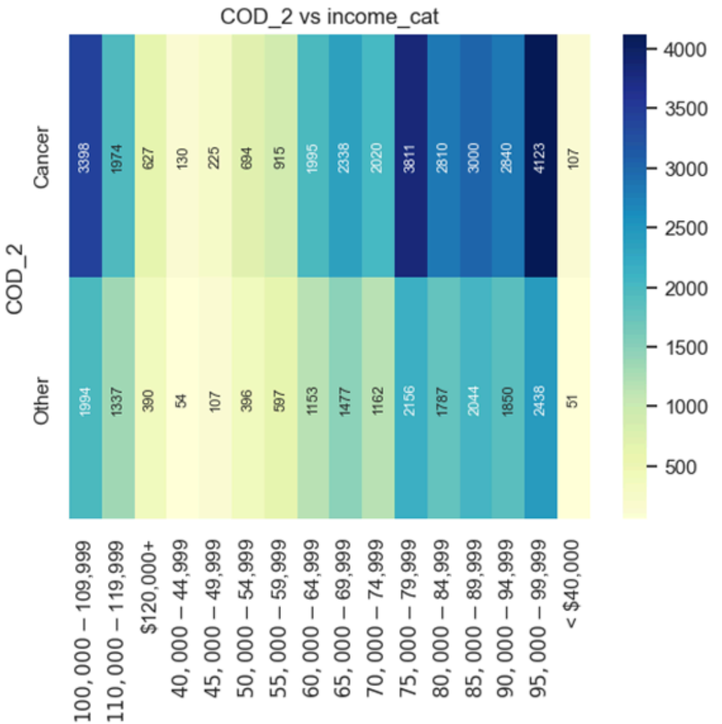
Demographics Vs COD

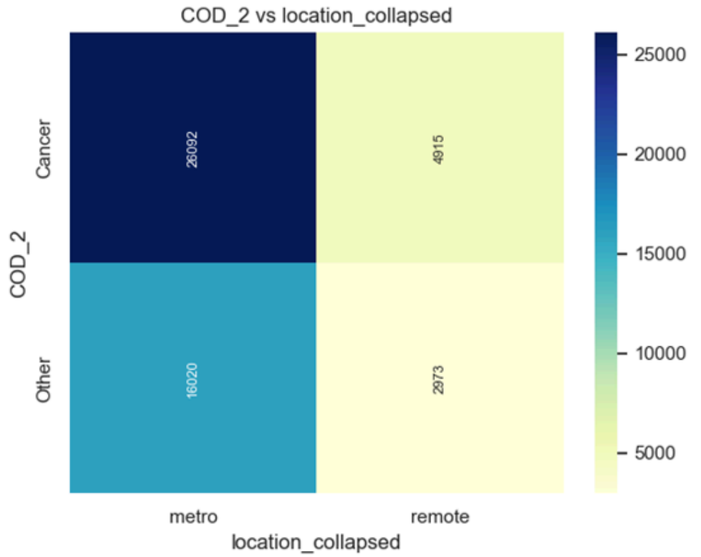




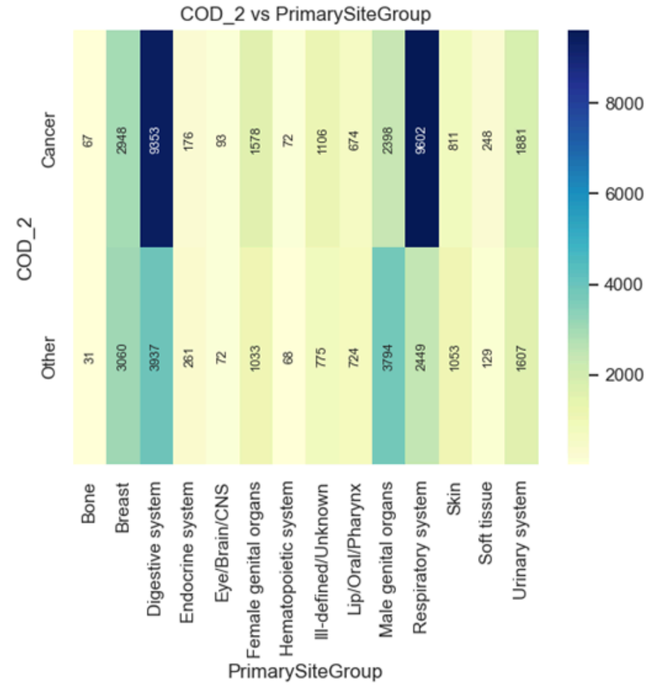


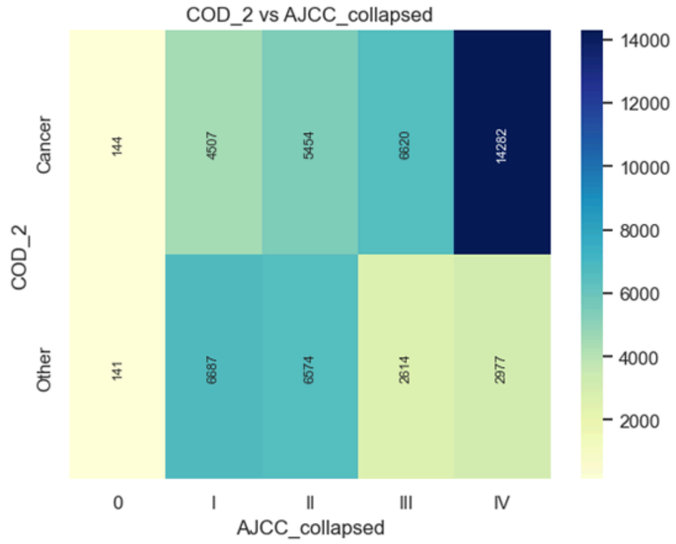


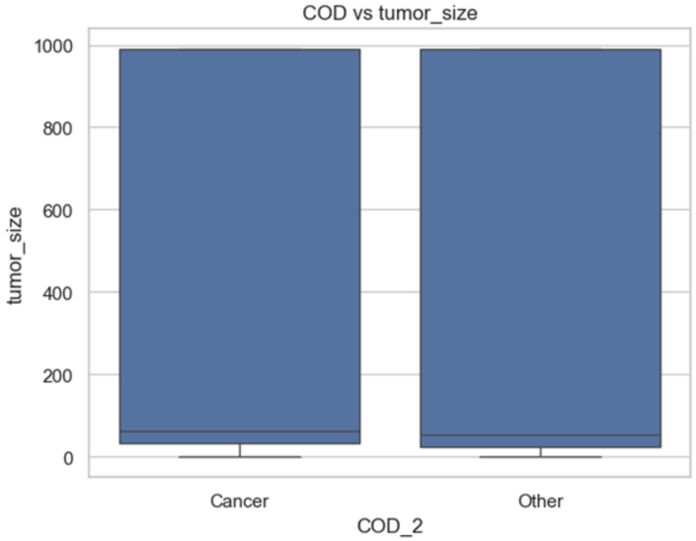


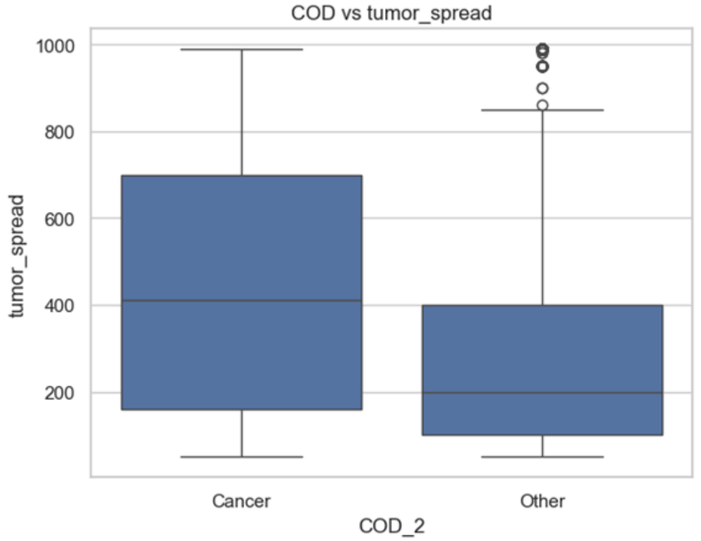


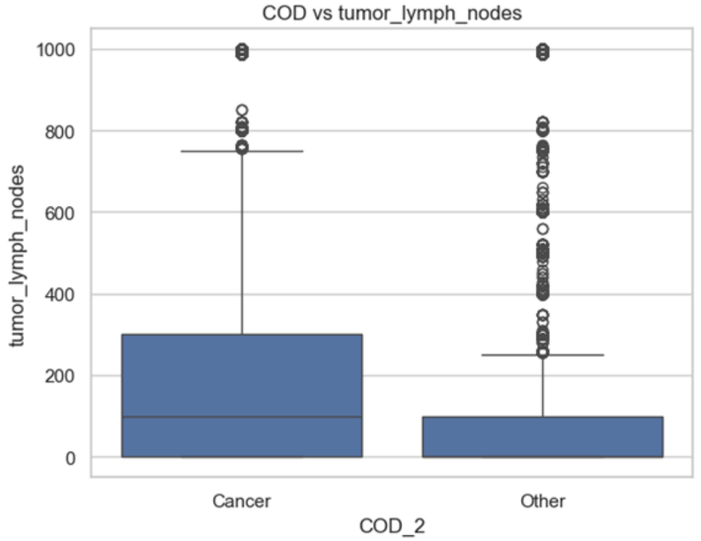
Tumor Characteristics Vs COD



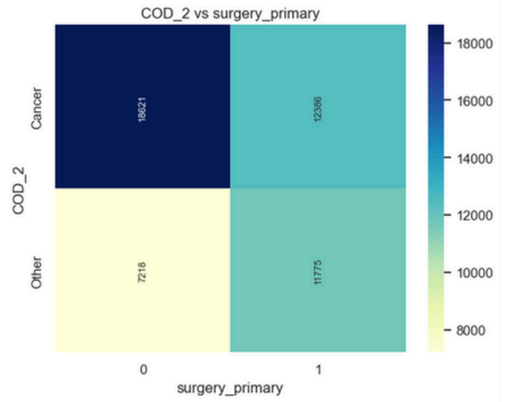


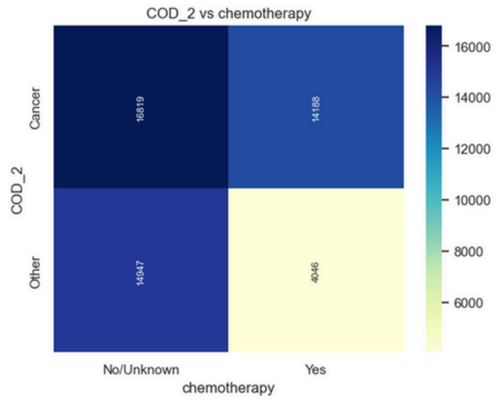


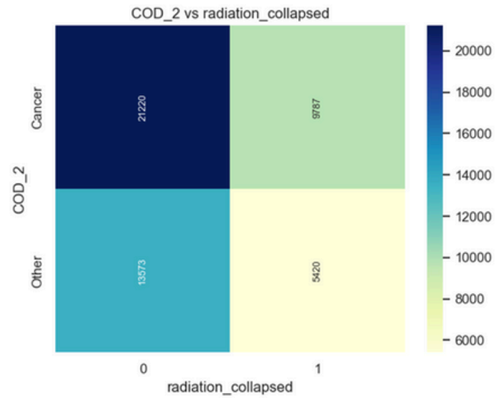




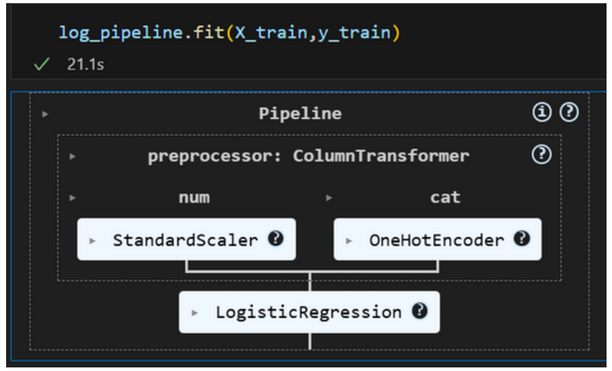
Treatments Vs COD







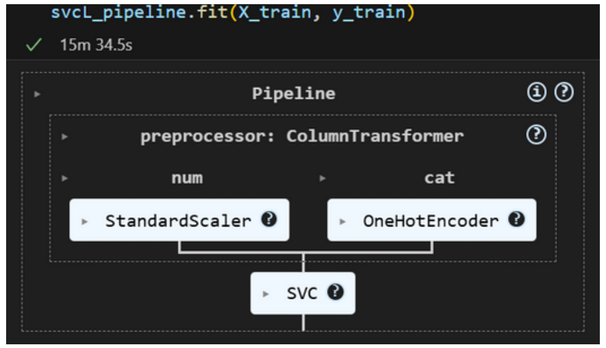
Model training

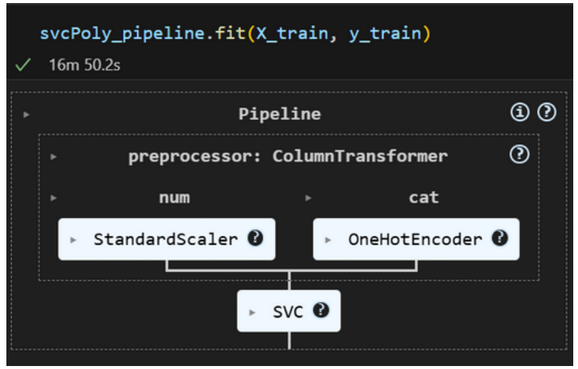


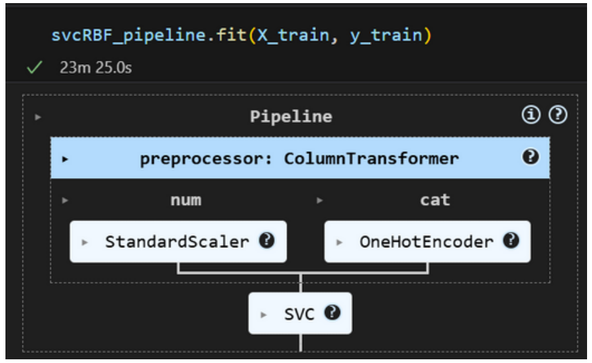

```
knn_grid = GridSearchCV(knn_pipeline, knn_param_grid, cv=3, scoring='roc_auc')
knn_grid.fit(X_train, y_train)
print("KNN best params:", knn_grid.best_params_)
print("KNN best CV ROC-AUC:", knn_grid.best_score_)
```

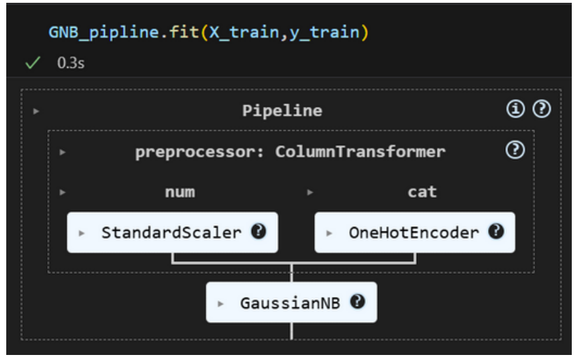
33.9s

```
KNN best params: {'knn__n_neighbors': 7}
KNN best CV ROC-AUC: 0.7340431687718653
```

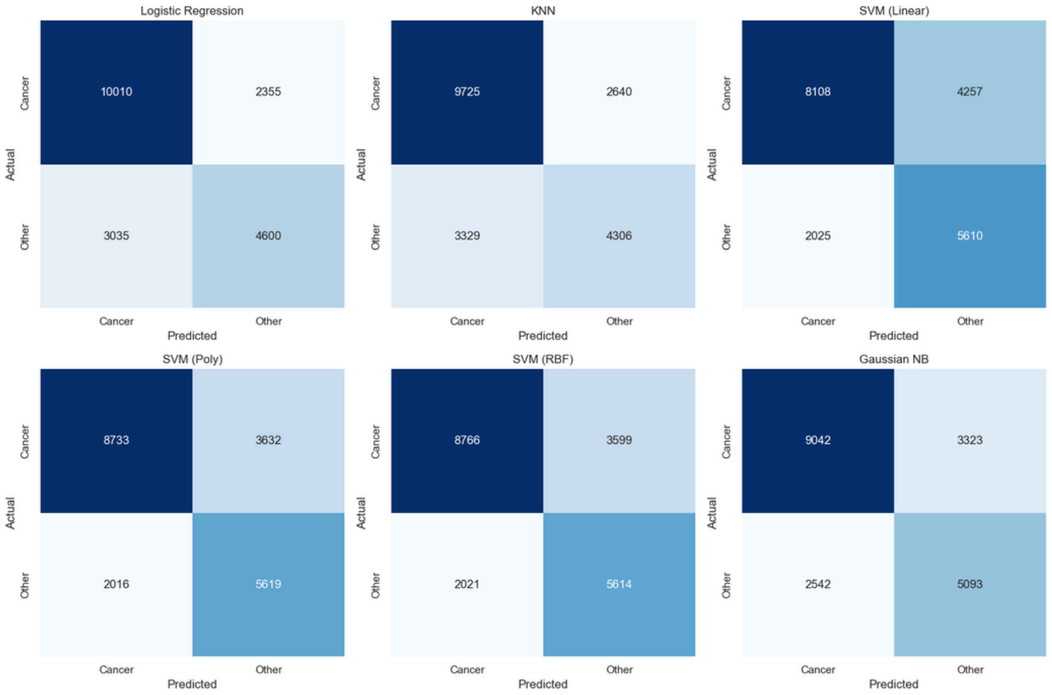


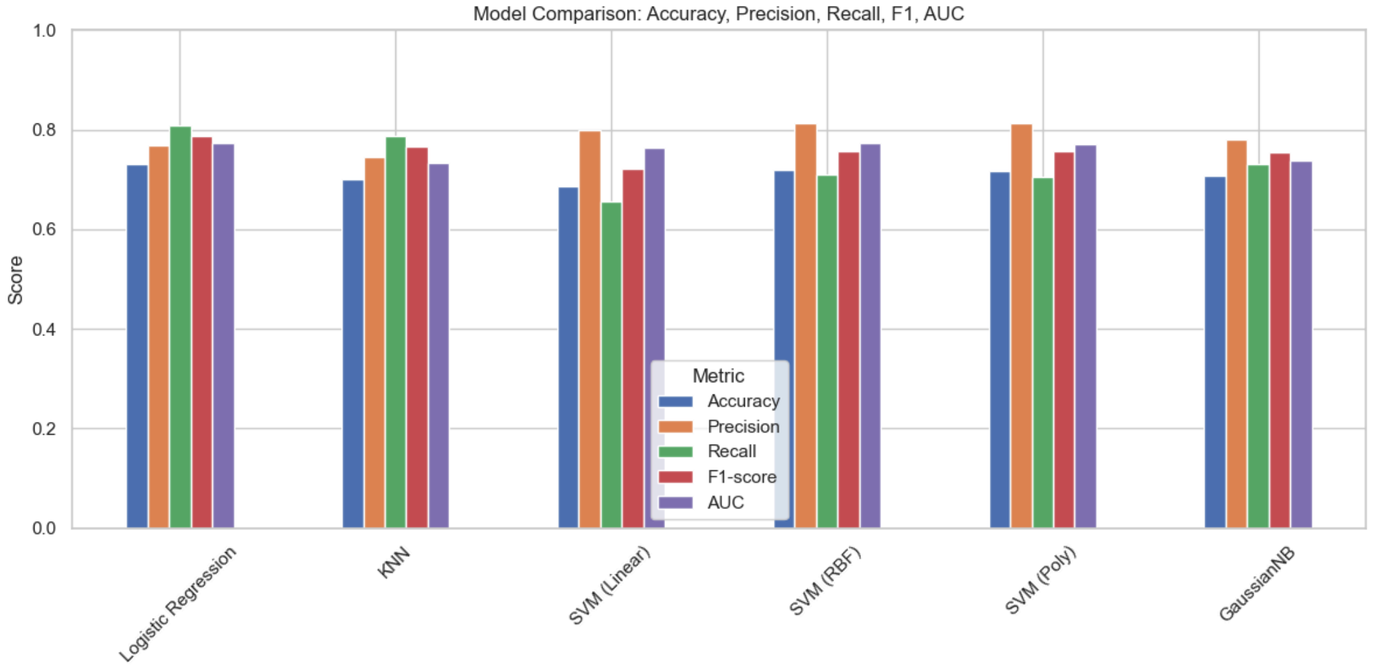


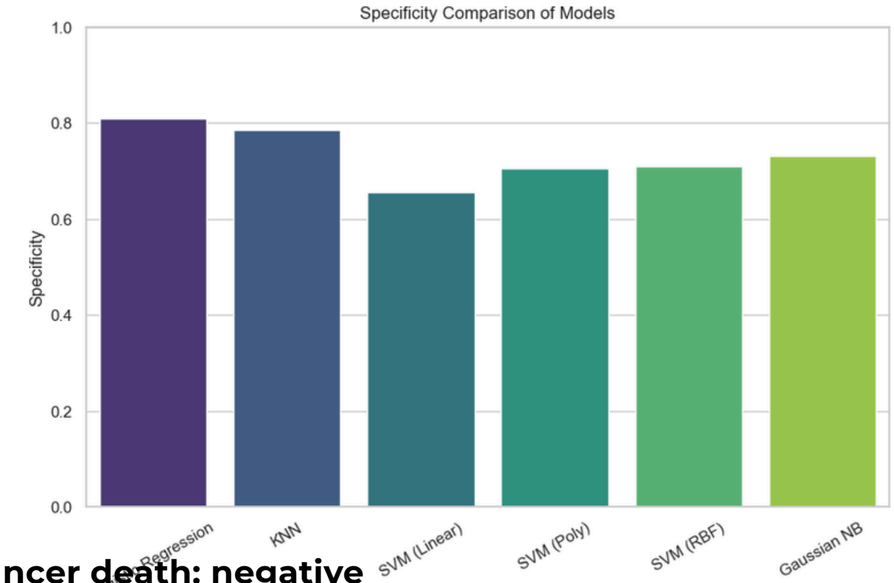




Evaluation







	Correct	Wrong
Logistic Regression	14610	5390
KNN	14031	5969
SVM (Linear)	13718	6282
SVM (RBF)	14380	5620
SVM (Poly)	14352	5648
GaussianNB	14135	5865
full	401503	152564

```
Precision: 0.764
Recall (Specificity): 0.802
F1-score: 0.783
Accuracy: 0.725
```

Cancer death: negative
other death: positive

Challenges

1. Data Preprocessing

- **Large, complex datasets with missing or inconsistent values.**

2. Computational Constraints

- **Very large dataset → slow training of models.**

3. Domain Knowledge

- **Limited understanding of clinical factors influencing survival and cause of death.**
- **Difficulty in feature selection without expert guidance.**

4. Wide scope and time constraints

Conclusion and Future work

Module 1

Logistic Regression effectively predicts patient survival on a 50k-sample, achieving 77% accuracy and 0.84 ROC-AUC, with fast, interpretable performance that can be further improved through feature engineering or tuning.

Module 2

Future work includes incorporating time since diagnosis and treatment duration patterns, and validating the model on European, Asian, and other populations to improve generalizability.

Module 3

Metrics are acceptable on 50K samples, but full 500K training will be done in the future to improve FN and specificity using models like Random Forest, XGBoost, and Neural Networks.

HTETOPS

THANK YOU

30, AUG, 2025