Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.

```
import os
os.environ["OPENBLAS_NUM_THREADS"] = "1"
import warnings
warnings.filterwarnings('ignore')

# Core imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import shap
import joblib

# Scikit-learn
from sklearn.model_selection import train_test_split, GridSearchCV, StratifiedKFold
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.metrics import (
    classification_report, confusion_matrix,
    roc_curve, auc, precision_recall_curve, average_precision_score
)
from sklearn.ensemble import RandomForestClassifier, StackingClassifier
from sklearn.svm import SVC

# XGBoost
from xgboost import XGBClassifier, plot_importance

# Imbalanced data
from imblearn.over_sampling import SMOTENC

from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures

# Colab file uploader
from google.colab import files

# 1) Load the data
```

```
uploaded = files.upload()
file_path = list(uploaded.keys())[0]
df = pd.read_csv(file_path)

# 2) Create survival targets
df["0.5yr_survival"] = (df["Overall survival [OS] (days)"] >= 180).astype(int)
df["1yr_survival"]  = (df["Overall survival [OS] (days)"] >= 365).astype(int)
df["1.5yr_survival"] = (df["Overall survival [OS] (days)"] >= 545).astype(int)
df["2yr_survival"]  = (df["Overall survival [OS] (days)"] >= 730).astype(int)

# 3) Quick plots of class balance
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
sns.countplot(data=df, x="1yr_survival", palette="Set2")
plt.xticks([0,1], ['<1yr','≥1yr']); plt.title("1yr Survival")
plt.subplot(1, 2, 2)
sns.countplot(data=df, x="2yr_survival", palette="Set1")
plt.xticks([0,1], ['<2yr','≥2yr']); plt.title("2yr Survival")
plt.tight_layout(); plt.show()

# 4) Feature & target lists
features = [
    'Age', 'Sex', 'Preoperative KPS', 'Previous treatment',
    'Histopathological subtype', 'WHO grade', 'IDH status',
    'Operative adjuncts', 'Preoperative contrast enhancing tumor volume (cm3)',
    'Extent of resection [EOR] %', 'EOR Category',
    'Adjuvant therapy', 'Radiotherapy treatment details',
    'Postoperative KPS', 'Postoperative Neurological Deficit'
]
targets = ["0.5yr_survival","1yr_survival","1.5yr_survival","2yr_survival"]

# To store results
base_auc_dict = {}
stacking_auc_dict = {}
best_params_dict = {}

def train_evaluate_model(target):
    print(f"\n=== Training & Evaluating: {target} ===")
    X = df[features]
    y = df[target]
```

```
    # 1) Train/test split
    X_train_raw, X_test_raw, y_train, y_test = train_test_split(
        X, y, test_size=0.2, stratify=y, random_state=42
    )

    # 2) Preprocessing: one-hot categorical, scale numeric
    cat_cols = X_train_raw.select_dtypes(include='object').columns.tolist()
    num_cols = X_train_raw.select_dtypes(include=np.number).columns.tolist()
    preprocessor = ColumnTransformer([
        ('cat', OneHotEncoder(handle_unknown='ignore', sparse_output=False), cat_cols),
        ('num', StandardScaler(), num_cols)
    ])

    X_train_enc = preprocessor.fit_transform(X_train_raw)
    X_test_enc  = preprocessor.transform(X_test_raw)

    # 3) Balance via SMOTENC (only on train)
    n_cat_feats = preprocessor.named_transformers_['cat'].get_feature_names_out().shape[0]
    smote = SMOTENC(categorical_features=list(range(n_cat_feats)), random_state=42)
    X_res, y_res = smote.fit_resample(X_train_enc, y_train)

    # 4) Feature selection: top 15 by ANOVA F-value
    selector = SelectKBest(f_classif, k=min(15, X_res.shape[1]))
    X_res_sel  = selector.fit_transform(X_res, y_res)
    X_test_sel = selector.transform(X_test_enc)

    # 5) Define model grid
    models = {
        'XGB': (XGBClassifier(eval_metric='logloss', use_label_encoder=False, random_state=42), {
            'n_estimators': [100, 200],
            'max_depth':    [3, 4, 5],
            'learning_rate':[0.01, 0.05, 0.1],
            'subsample':    [0.8, 1.0]
        }),
        'RF': (RandomForestClassifier(random_state=42), {
            'n_estimators': [100, 200],
            'max_depth':    [4, 5]
        }),
    'SVM': (Pipeline([
            ('poly', PolynomialFeatures(degree=2, interaction_only=True, include_bias=False)),
```

```
            ('svc', SVC(probability=True, class_weight='balanced', random_state=42))
        ]), {
            'svc__C': [0.1, 1, 10],
            'svc__kernel': ['rbf', 'poly']
        })
    }

    best_estimators = {}
    base_roc = {}
    base_auc = {}
    cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

    for name, (mdl, param_grid) in models.items():
        grid = GridSearchCV(mdl, param_grid, scoring='roc_auc', cv=cv, n_jobs=-1, verbose=1)
        grid.fit(X_res_sel, y_res)
        best = grid.best_estimator_
        best_estimators[name] = best
        best_params_dict[f"{target}_{name}"] = grid.best_params_
        proba = best.predict_proba(X_test_sel)[:,1]
        fpr_i, tpr_i, _ = roc_curve(y_test, proba)
        auc_i = auc(fpr_i, tpr_i)
        base_roc[name] = (fpr_i, tpr_i)
        base_auc[name] = auc_i
        print(f"  → {name}: best_params={grid.best_params_}, AUC={auc_i:.3f}")

    # 6) Stacking ensemble
    stack = StackingClassifier(
        estimators    = list(best_estimators.items()),
        final_estimator = RandomForestClassifier(n_estimators=100, random_state=42),
        cv = cv, n_jobs = -1
    )
    stack.fit(X_res_sel, y_res)
    # compute stacking AUC
    proba_s = stack.predict_proba(X_test_sel)[:,1]
    fpr_s, tpr_s, _ = roc_curve(y_test, proba_s)
    auc_s = auc(fpr_s, tpr_s)
    print(f"  → Stacking AUC = {auc_s:.3f}")

    # store results
    base_auc_dict[target] = base_auc
    stacking_auc_dict[target] = auc_s
```

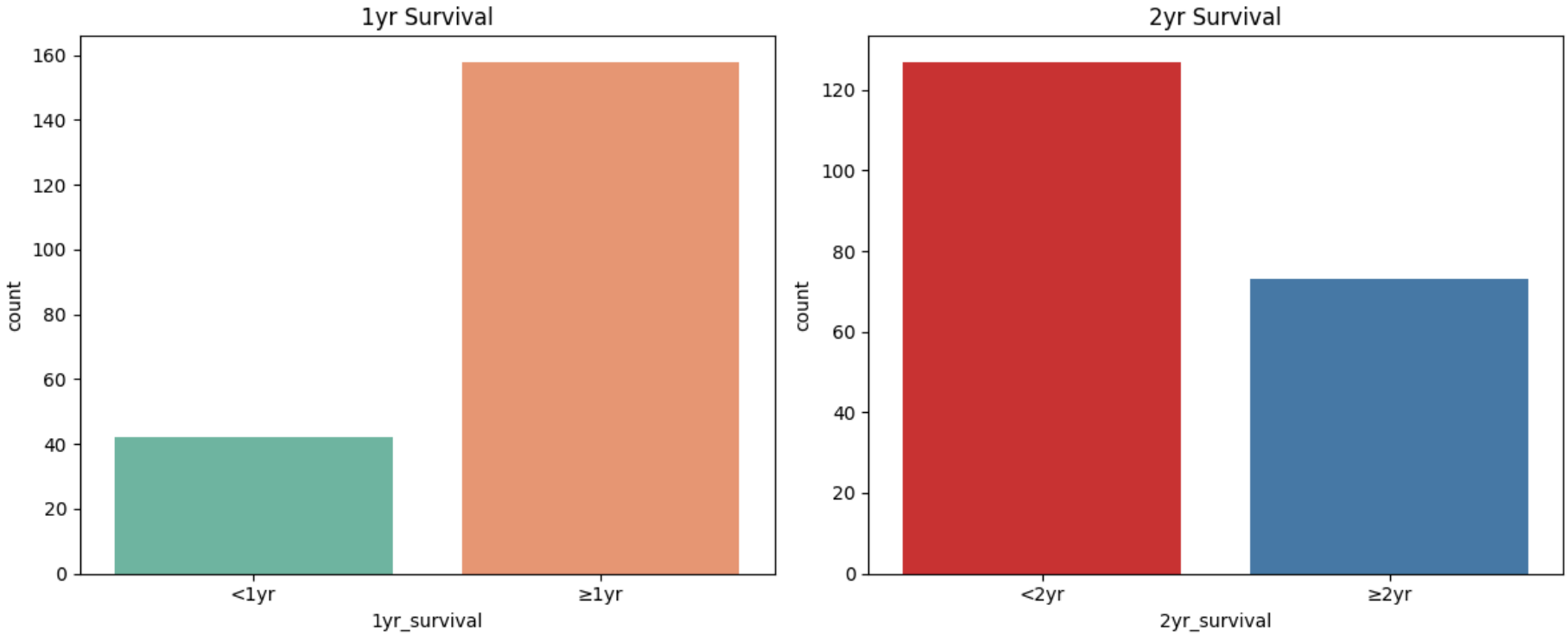
```
    stacking_auc_dict[target] = auc_s

    # 7) Plot all ROC curves
    plt.figure(figsize=(6,5))
    for name, (fpr_i, tpr_i) in base_roc.items():
        plt.plot(fpr_i, tpr_i, label=f"{name} (AUC={base_auc[name]:.3f})")
    plt.plot(fpr_s, tpr_s, label=f"Stacking (AUC={auc_s:.3f})", linewidth=2, linestyle='--')
    plt.plot([0,1],[0,1],'k--')
    plt.title(f'ROC Curves — {target}')
    plt.xlabel('FPR'); plt.ylabel('TPR')
    plt.legend(); plt.grid(True); plt.tight_layout(); plt.show()

    # 8) Classification report & confusion at 0.5
    y_pred = (proba_s >= 0.5).astype(int)
    print(classification_report(y_test, y_pred))
    cm = confusion_matrix(y_test, y_pred)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                xticklabels=['Neg','Pos'], yticklabels=['Neg','Pos'])
    plt.title('Confusion Matrix (Stacking)'); plt.xlabel('Pred'); plt.ylabel('Actual')
    plt.tight_layout(); plt.show()

    # 9) Precision-Recall (stacking)
    pr_auc = average_precision_score(y_test, proba_s)
    precisions, recalls, _ = precision_recall_curve(y_test, proba_s)
    plt.figure(figsize=(6,5))
    plt.plot(recalls, precisions, label=f'PR AUC = {pr_auc:.3f}')
    plt.title(f'Precision-Recall — {target}')
    plt.xlabel('Recall'); plt.ylabel('Precision')
    plt.legend(); plt.grid(True); plt.tight_layout(); plt.show()

    # 10) SHAP on best XGB
    explainer = shap.Explainer(best_estimators['XGB'])
    shap_vals = explainer(X_test_sel)
    feat_names = selector.get_feature_names_out(preprocessor.get_feature_names_out())
    X_test_df = pd.DataFrame(X_test_sel, columns=feat_names)

    plt.figure(figsize=(8,6))
    shap.summary_plot(shap_vals, X_test_df, plot_type='bar', show=False)
    plt.title('SHAP Feature Importance (XGB)'); plt.tight_layout(); plt.show()

    plt.figure(figsize=(8,6))
```

```
    shap.summary_plot(shap_vals, X_test_df, show=False)
    plt.title('SHAP Beeswarm (XGB)'); plt.tight_layout(); plt.show()

    # 11) Save models
    joblib.dump(best_estimators['XGB'], f"best_xgb_{target}.pkl")
    joblib.dump(stack,          f"stack_model_{target}.pkl")

# Run for each target
for t in targets:
    train_evaluate_model(t)

# Summary of AUCs
print("\n=== Base Model AUCs by Target ===")
for t, aucs in base_auc_dict.items():
    print(f"{t}: " + ", ".join([f"{name}={auc:.3f}" for name, auc in aucs.items()]))
print("\n=== Stacking AUCs ===")
for t, auc in stacking_auc_dict.items():
    print(f"{t}: {auc:.3f}")

# Optional: overall AUC comparison for stacking
plt.figure(figsize=(6,4))
sns.barplot(x=list(stacking_auc_dict.keys()), y=list(stacking_auc_dict.values()))
plt.ylim(0,1); plt.title("Stacking AUC by Target"); plt.ylabel("AUC")
plt.tight_layout(); plt.show()

# Kaplan–Meier analysis (optional)

from lifelines import KaplanMeierFitter

plt.figure(figsize=(8,6))
kmf = KaplanMeierFitter()
for grade, grp in df.groupby('WHO grade'):
    kmf.fit(grp["Overall survival [OS] (days)"]/365,
            grp["1yr_survival"], label=f"WHO Grade {grade}")
    kmf.plot_survival_function()
plt.title("Kaplan–Meier: 1yr Survival by WHO Grade")
plt.xlabel("Years"); plt.ylabel("Survival Probability")
plt.tight_layout(); plt.show()
```

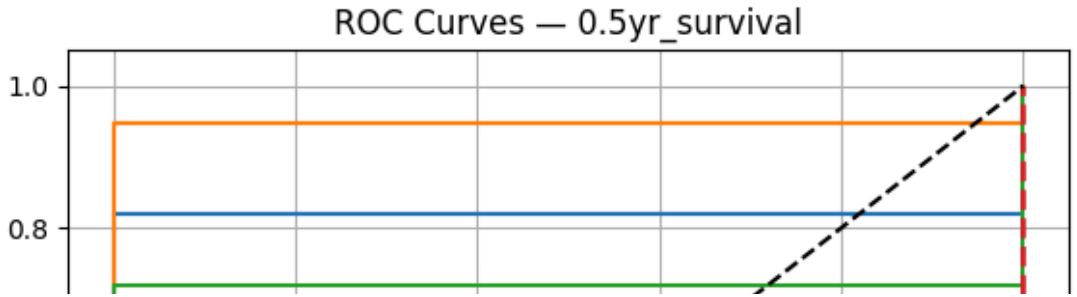
Choose Files patient_datamain.csv

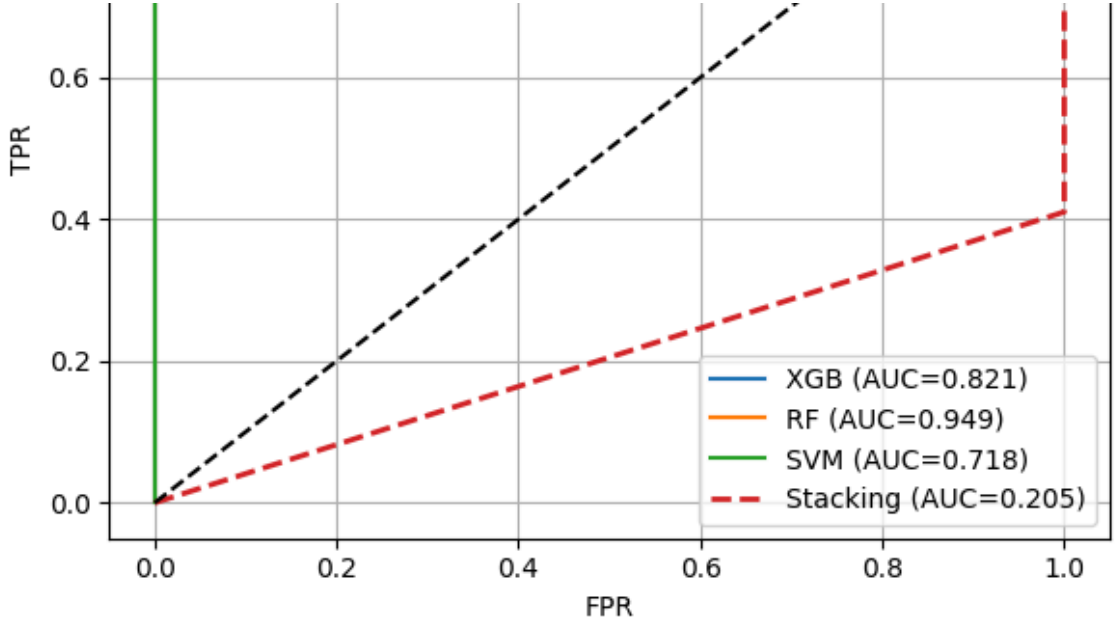
- **patient_datamain.csv**(text/csv) - 29974 bytes, last modified: 4/16/2025 - 100% done

```
Saving patient_datamain.csv to patient_datamain (3).csv
```

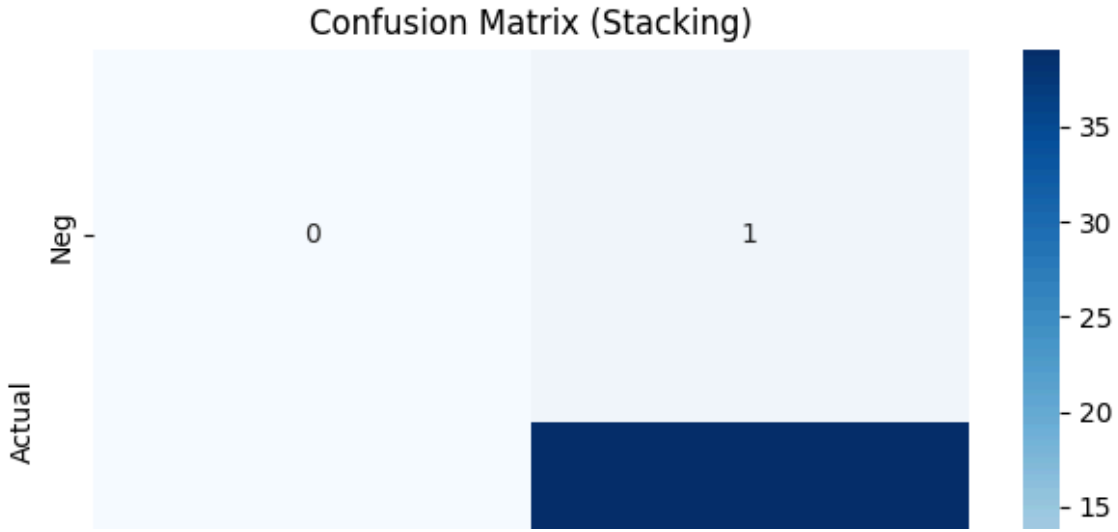


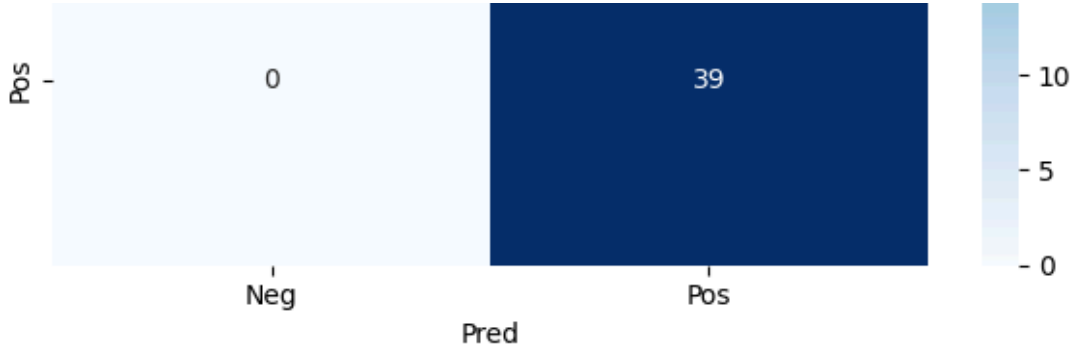

```
=== Training & Evaluating: 0.5yr_survival ===
Fitting 5 folds for each of 36 candidates, totalling 180 fits
  → XGB: best_params={'learning_rate': 0.05, 'max_depth': 3, 'n_estimators': 100, 'subsample': 0.8}, AUC=0.821
Fitting 5 folds for each of 4 candidates, totalling 20 fits
  → RF: best_params={'max_depth': 4, 'n_estimators': 200}, AUC=0.949
Fitting 5 folds for each of 6 candidates, totalling 30 fits
  → SVM: best_params={'svc__C': 0.1, 'svc__kernel': 'poly'}, AUC=0.718
  → Stacking AUC = 0.205
```

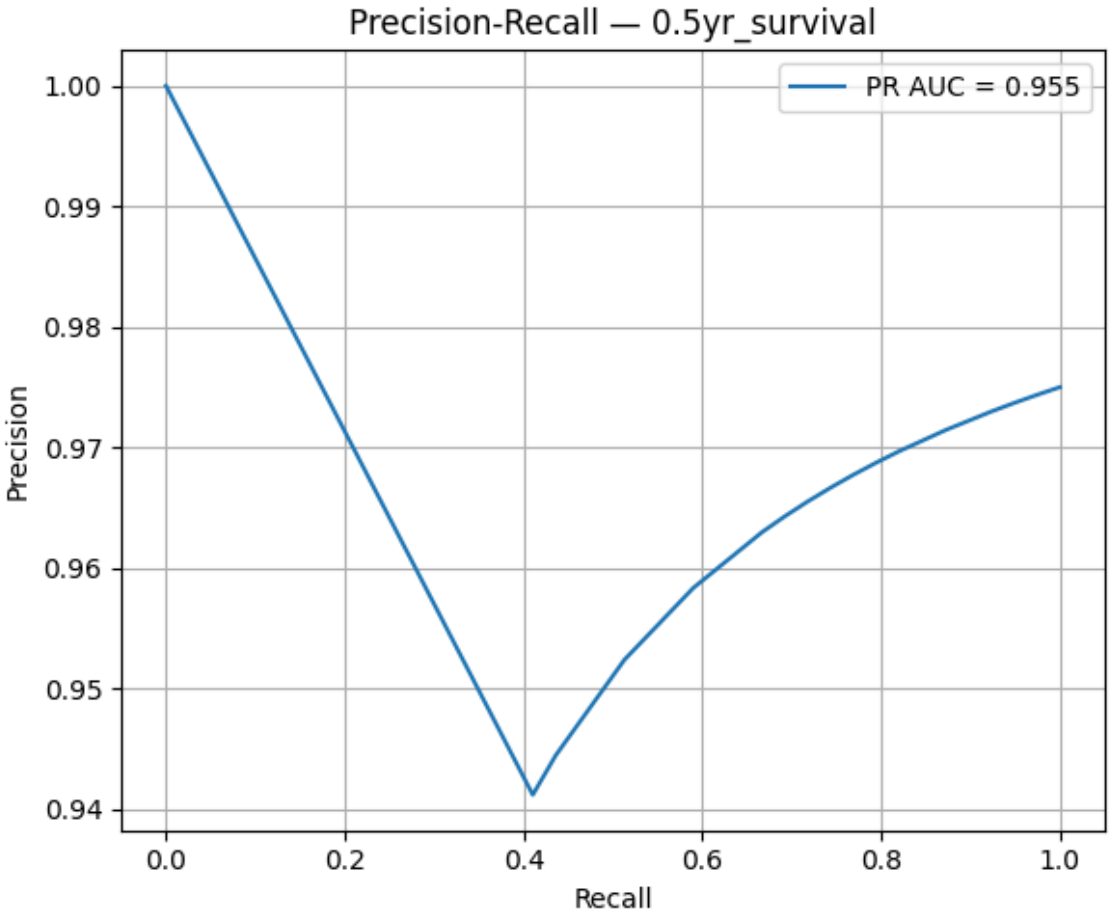


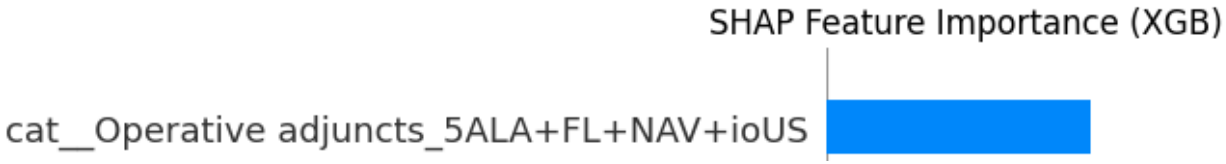


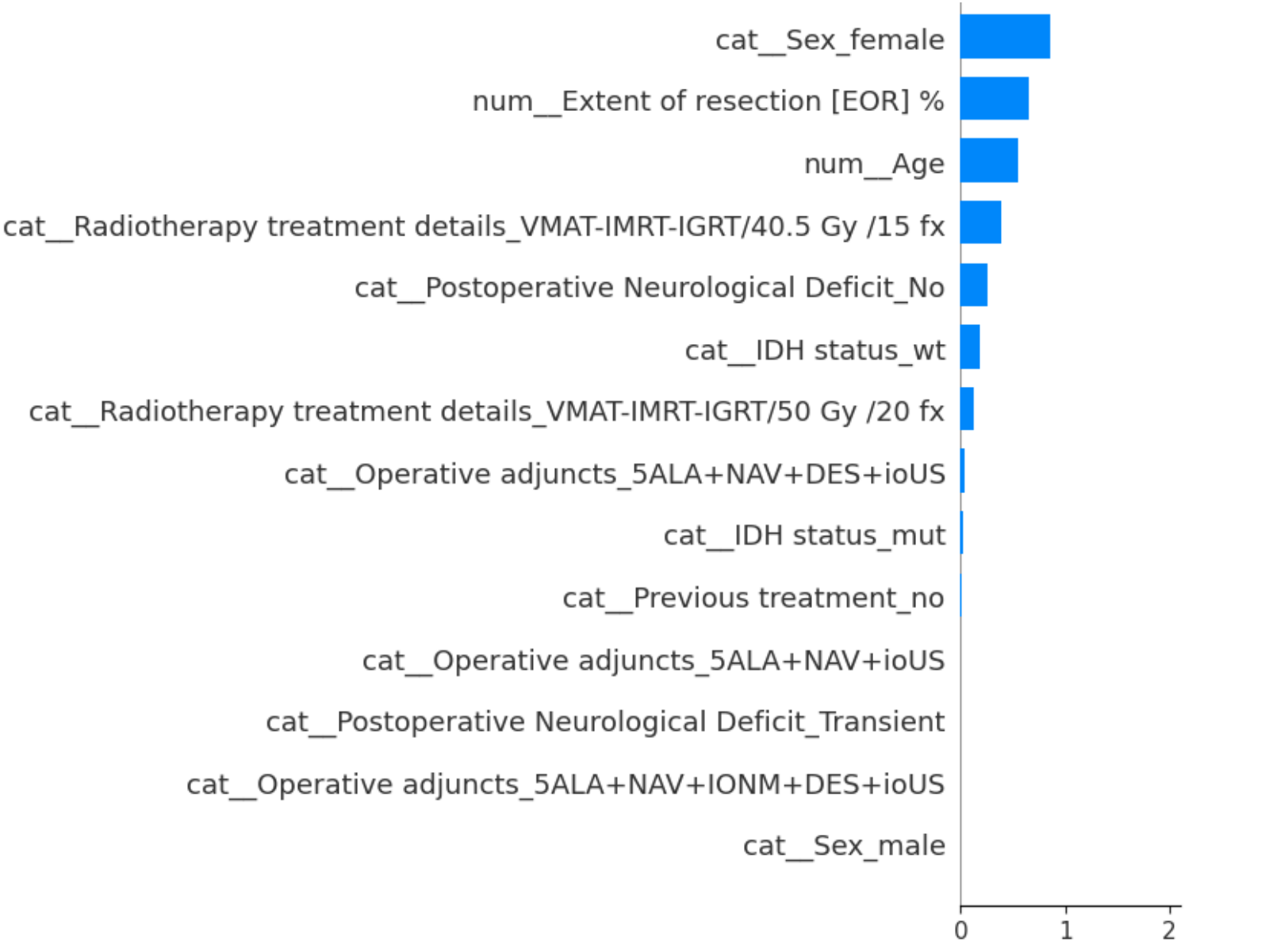
	precision	recall	f1-score	support
0	0.00	0.00	0.00	1
1	0.97	1.00	0.99	39
accuracy			0.97	40
macro avg	0.49	0.50	0.49	40
weighted avg	0.95	0.97	0.96	40

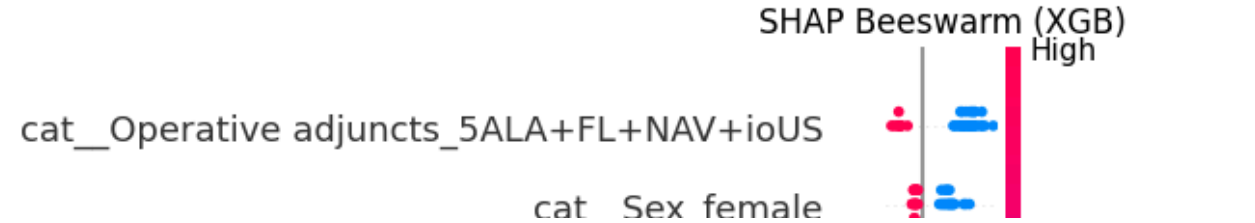


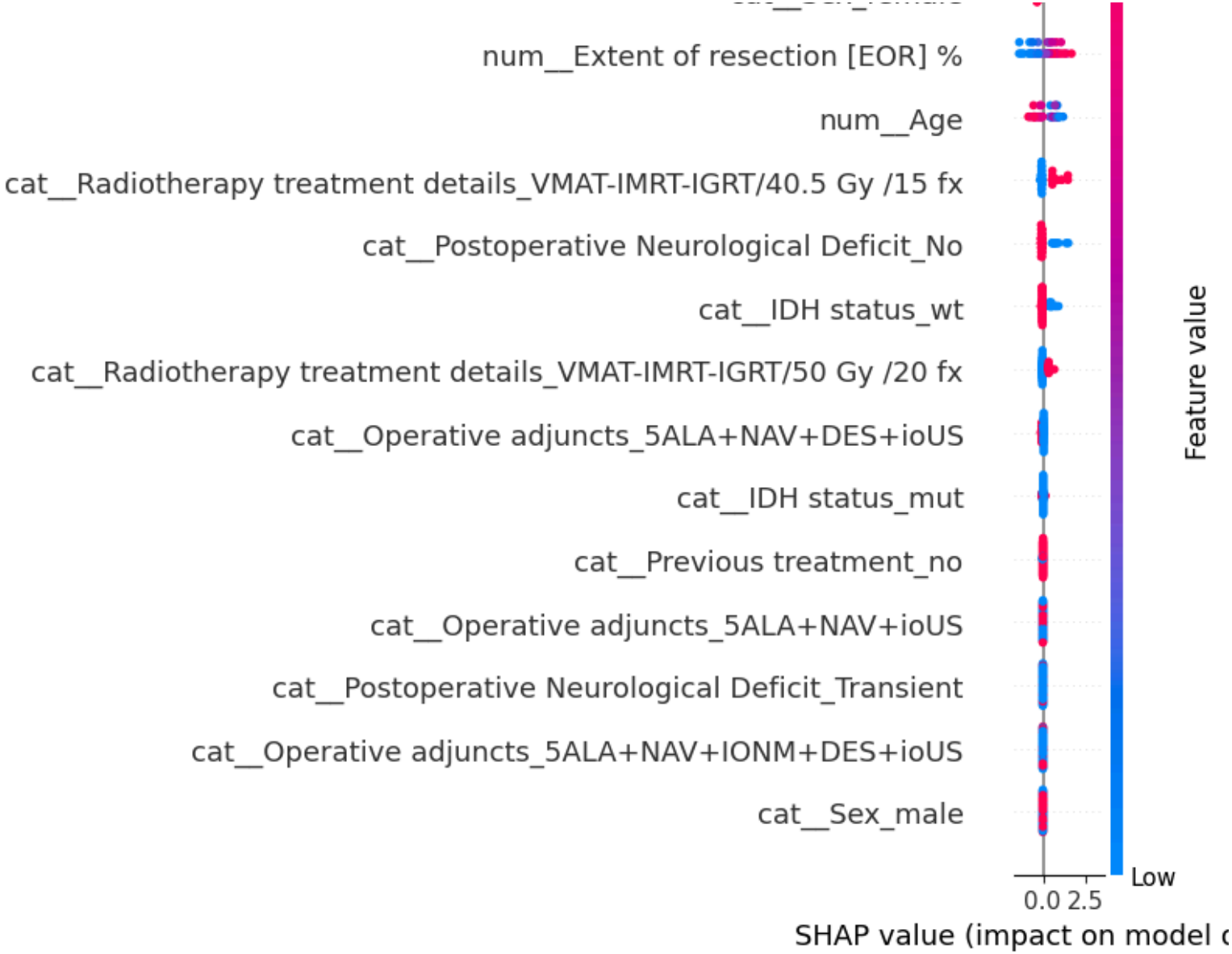

Pos
0
39
10
5
0
Neg
Pos
Pred


Precision-Recall — 0.5yr_survival
PR AUC = 0.955
1.00
0.99
0.98
0.97
0.96
0.95
0.94
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall

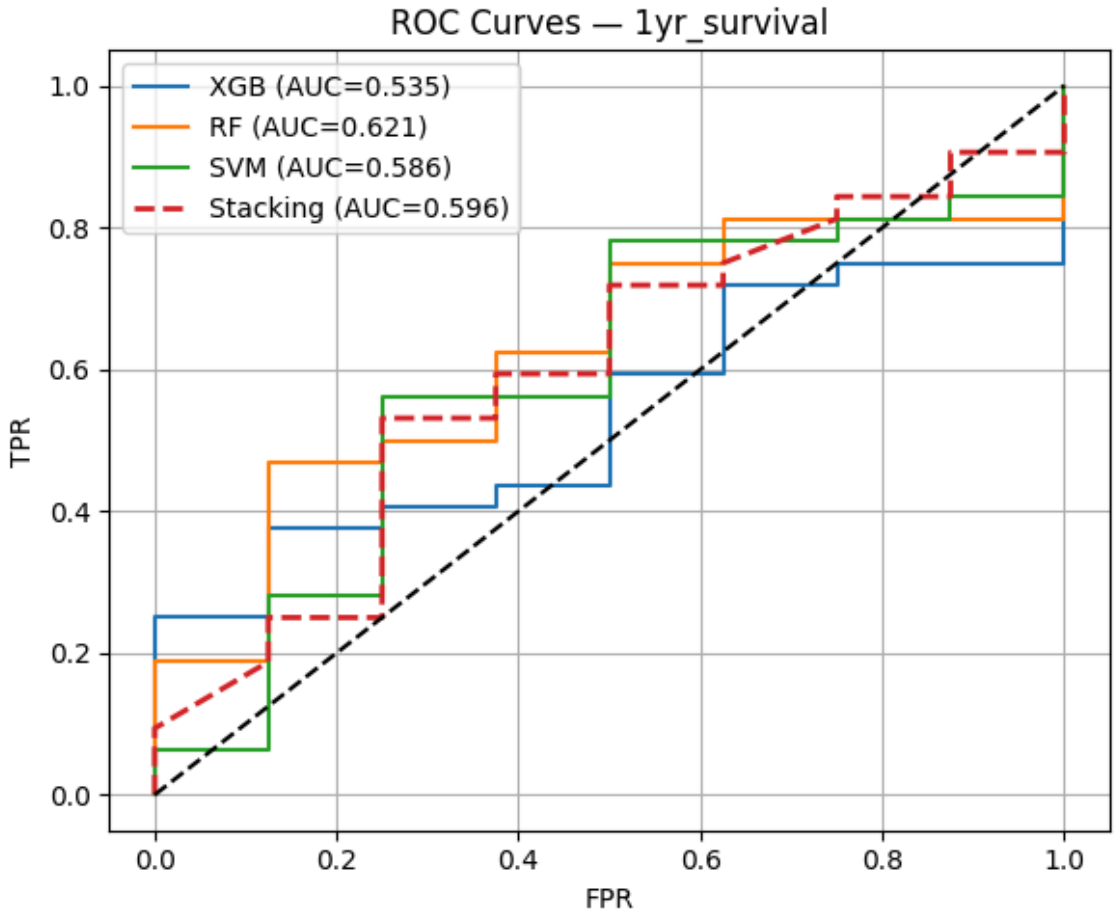

SHAP Feature Importance (XGB)
cat__Operative adjuncts_5ALA+FL+NAV+ioUS


cat__Sex_female
num__Extent of resection [EOR] %
num__Age
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/40.5 Gy /15 fx
cat__Postoperative Neurological Deficit_No
cat__IDH status_wt
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/50 Gy /20 fx
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__IDH status_mut
cat__Previous treatment_no
cat__Operative adjuncts_5ALA+NAV+ioUS
cat__Postoperative Neurological Deficit_Transient
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__Sex_male
0
1
2
mean(|SHAP value|) (average impact on model o

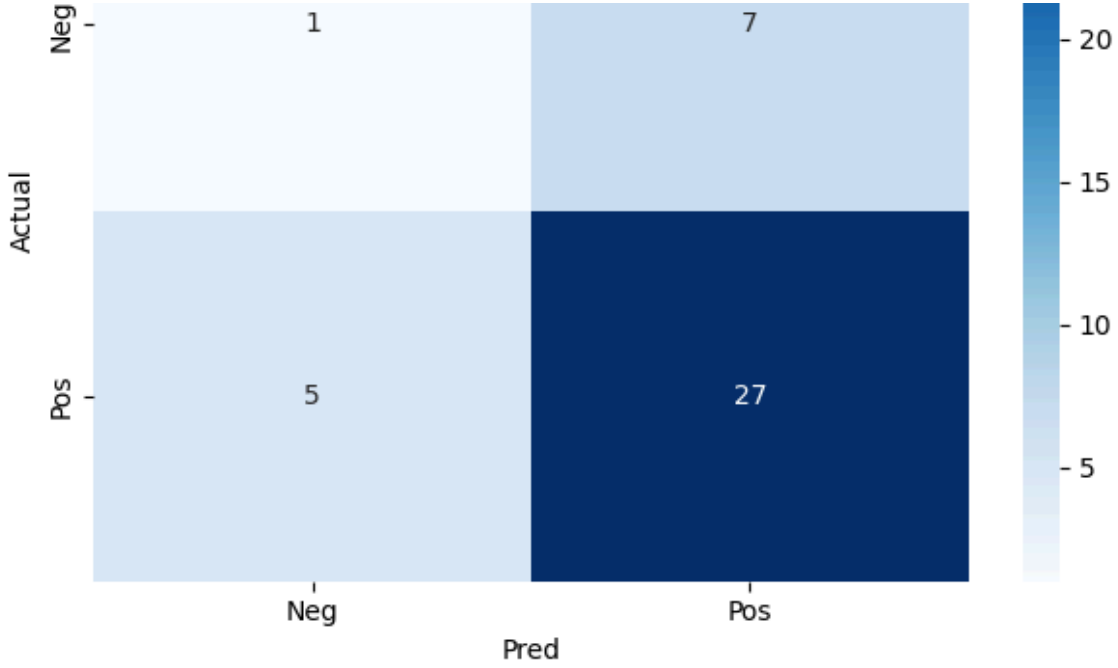

SHAP Beeswarm (XGB)
High
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__Sex_female

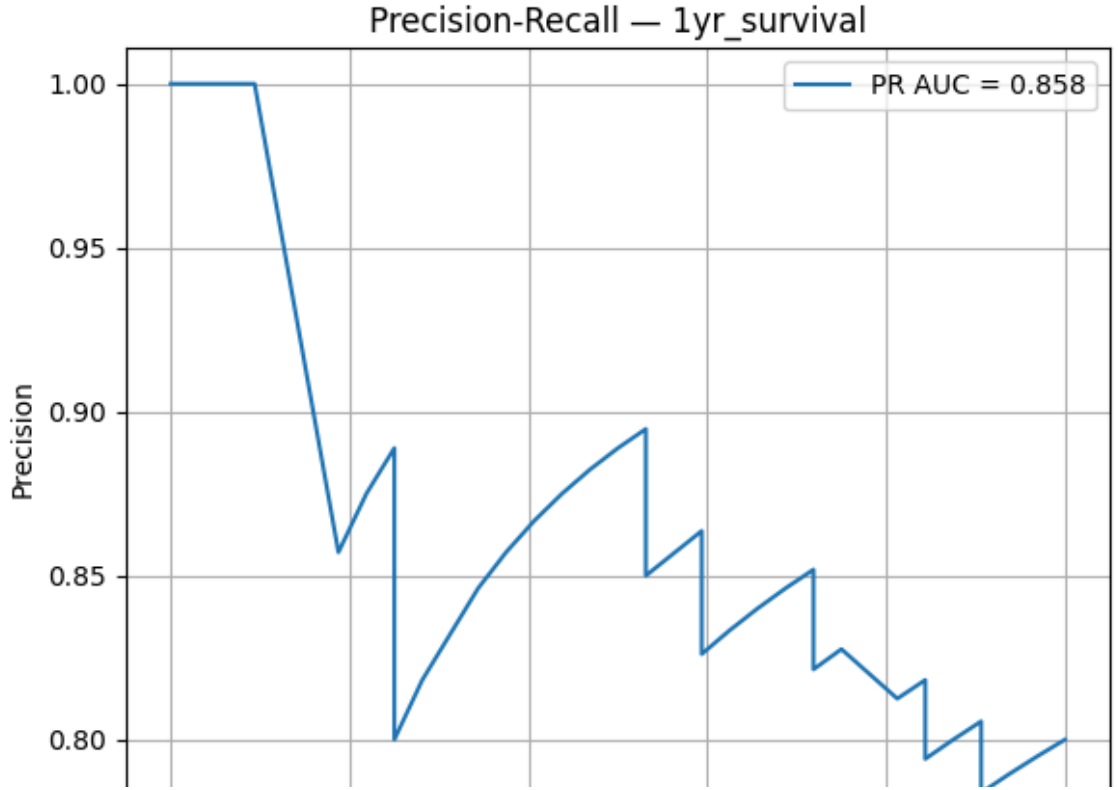


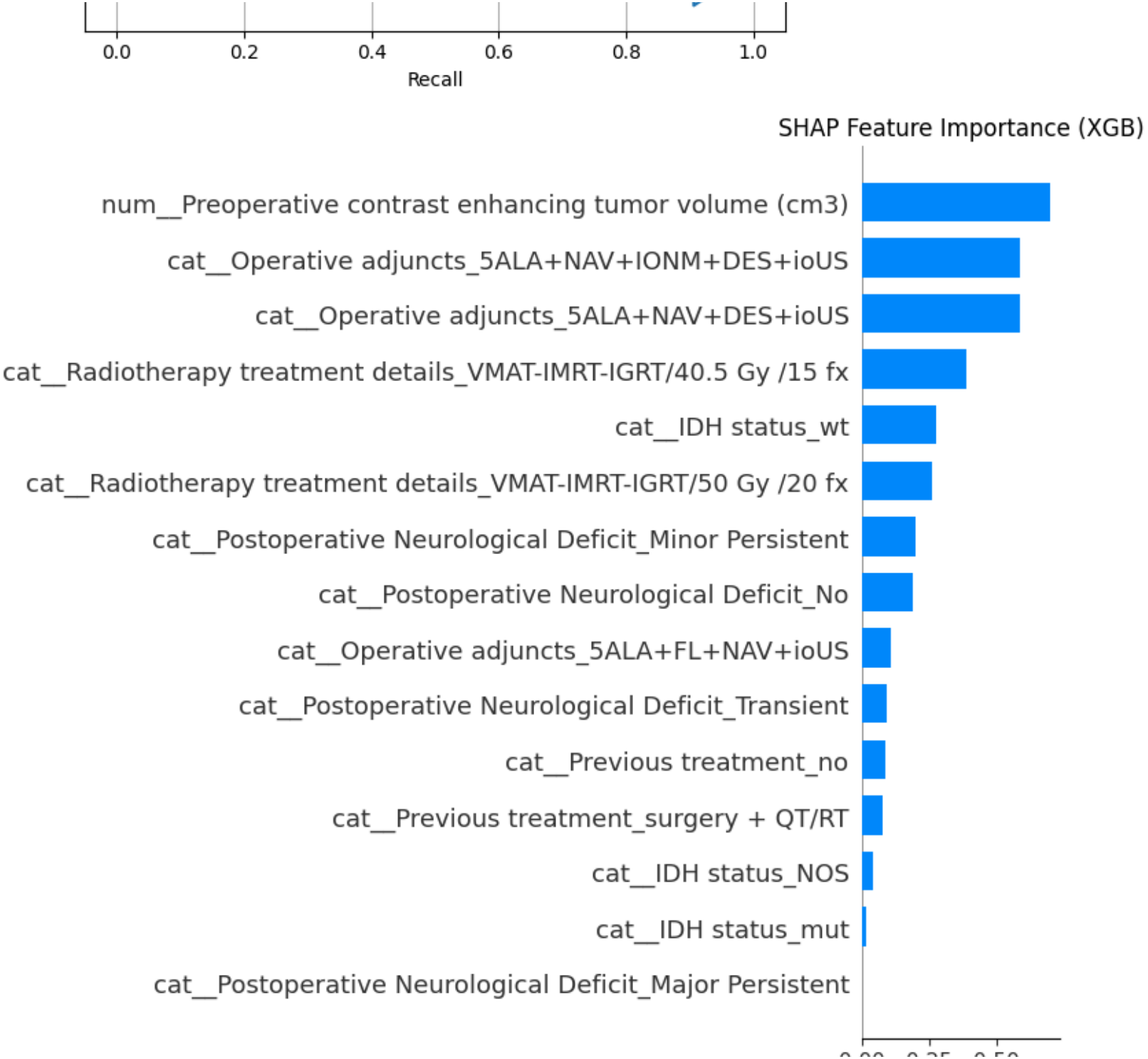

```
=== Training & Evaluating: 1yr_survival ===
Fitting 5 folds for each of 36 candidates, totalling 180 fits
  → XGB: best_params={'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'subsample': 1.0}, AUC=0.535
Fitting 5 folds for each of 4 candidates, totalling 20 fits
  → RF: best_params={'max_depth': 5, 'n_estimators': 200}, AUC=0.621
Fitting 5 folds for each of 6 candidates, totalling 30 fits
  → SVM: best_params={'svc__C': 1, 'svc__kernel': 'rbf'}, AUC=0.586
  → Stacking AUC = 0.596
```

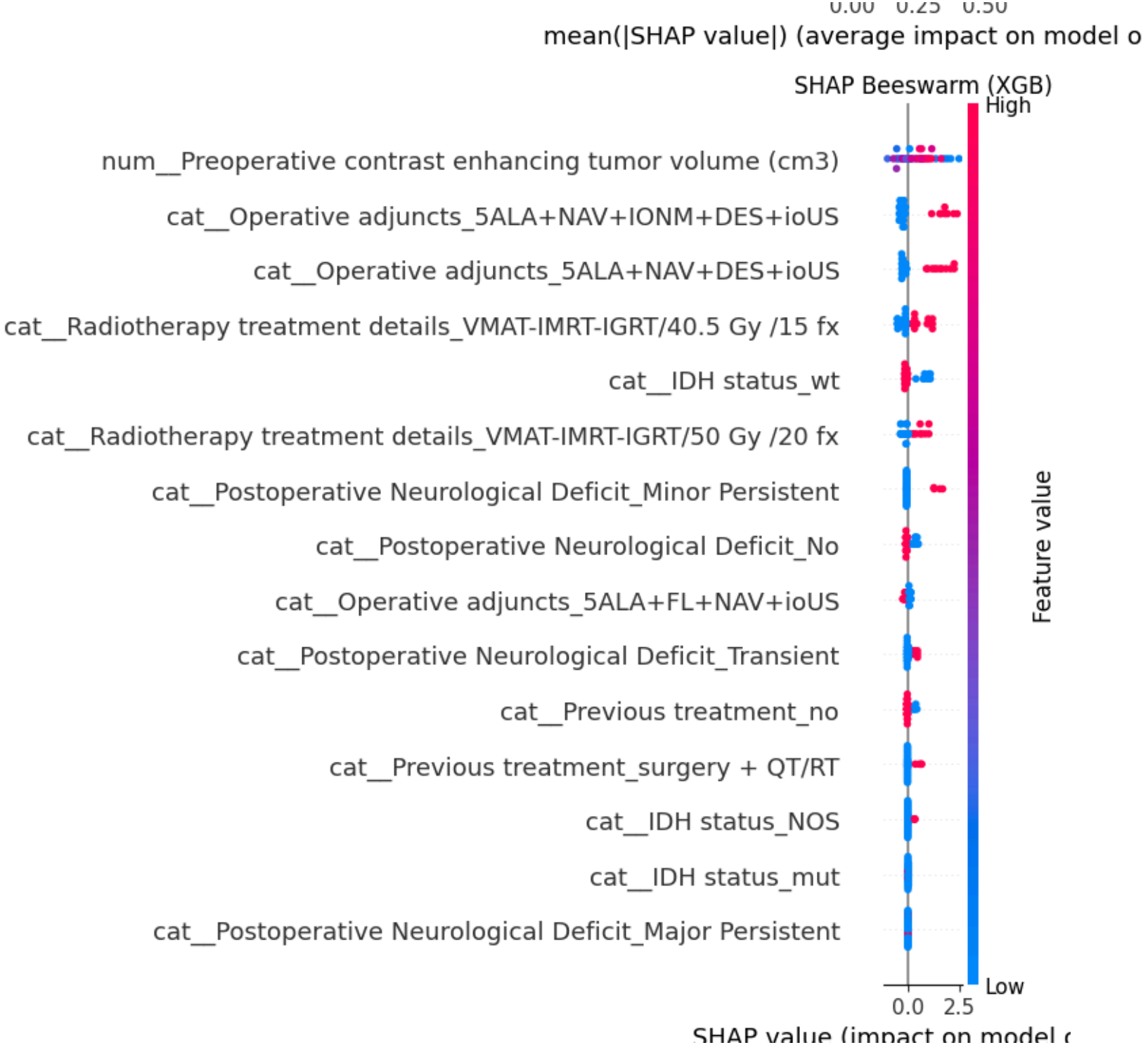


	precision	recall	f1-score	support
0	0.17	0.12	0.14	8
1	0.79	0.84	0.82	32
accuracy			0.70	40
macro avg	0.48	0.48	0.48	40
weighted avg	0.67	0.70	0.68	40

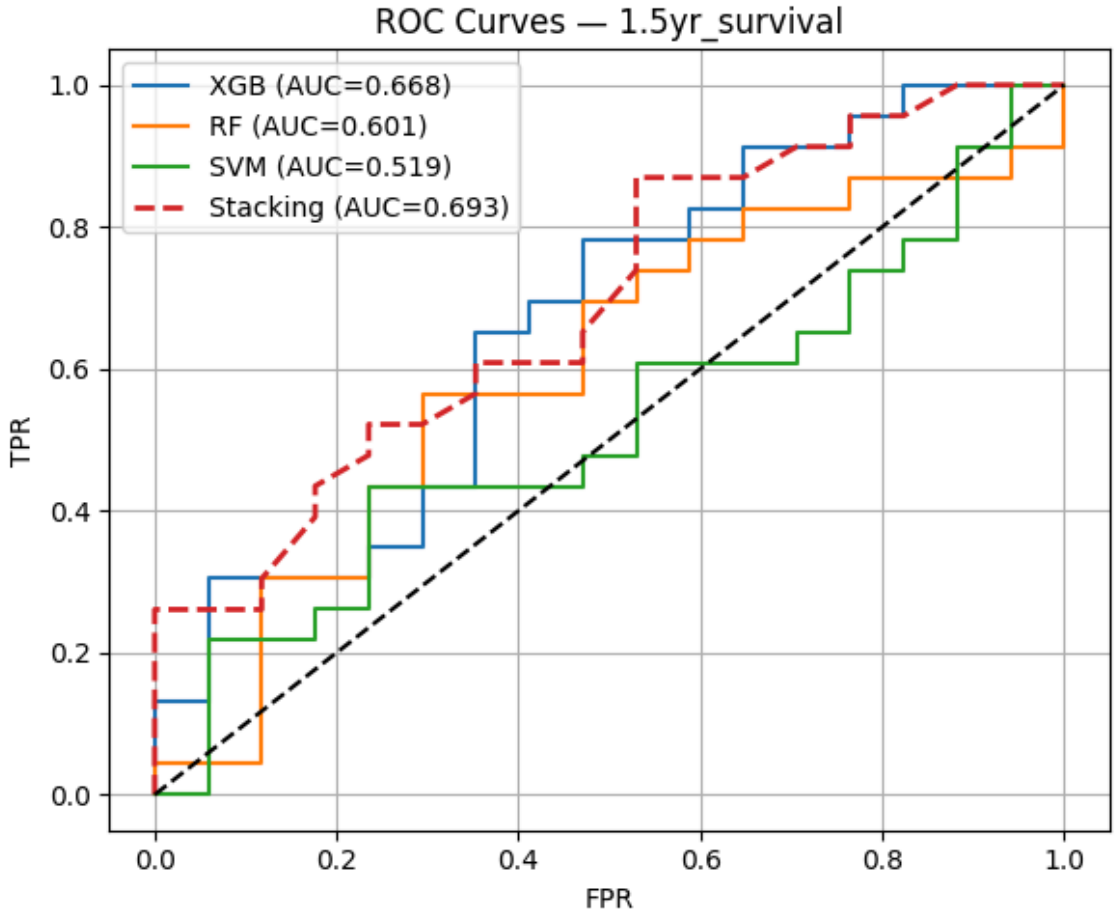

Neg
Pos
Actual
1
7
5
27
Neg
Pos
Pred
20
15
10
5


Precision-Recall — 1yr_survival
PR AUC = 0.858
Precision
1.00
0.95
0.90
0.85
0.80


0.0
0.2
0.4
0.6
0.8
1.0
Recall
SHAP Feature Importance (XGB)
num__Preoperative contrast enhancing tumor volume (cm3)
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/40.5 Gy /15 fx
cat__IDH status_wt
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/50 Gy /20 fx
cat__Postoperative Neurological Deficit_Minor Persistent
cat__Postoperative Neurological Deficit_No
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__Postoperative Neurological Deficit_Transient
cat__Previous treatment_no
cat__Previous treatment_surgery + QT/RT
cat__IDH status_NOS
cat__IDH status_mut
cat__Postoperative Neurological Deficit_Major Persistent

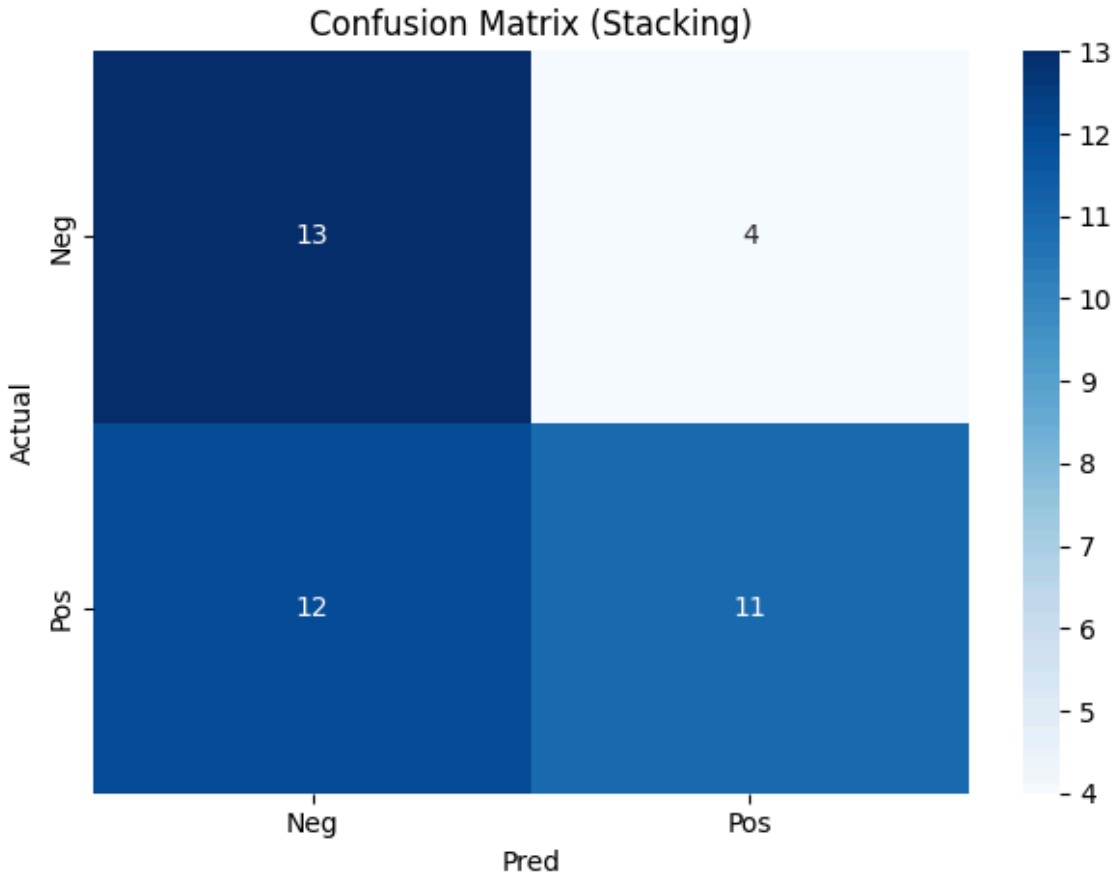

0.00 0.25 0.50
mean(|SHAP value|) (average impact on model o
SHAP Beeswarm (XGB)
num__Preoperative contrast enhancing tumor volume (cm3)
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/40.5 Gy /15 fx
cat__IDH status_wt
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/50 Gy /20 fx
cat__Postoperative Neurological Deficit_Minor Persistent
cat__Postoperative Neurological Deficit_No
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__Postoperative Neurological Deficit_Transient
cat__Previous treatment_no
cat__Previous treatment_surgery + QT/RT
cat__IDH status_NOS
cat__IDH status_mut
cat__Postoperative Neurological Deficit_Major Persistent
High
Feature value
Low
0.0 2.5
SHAP value (impact on model

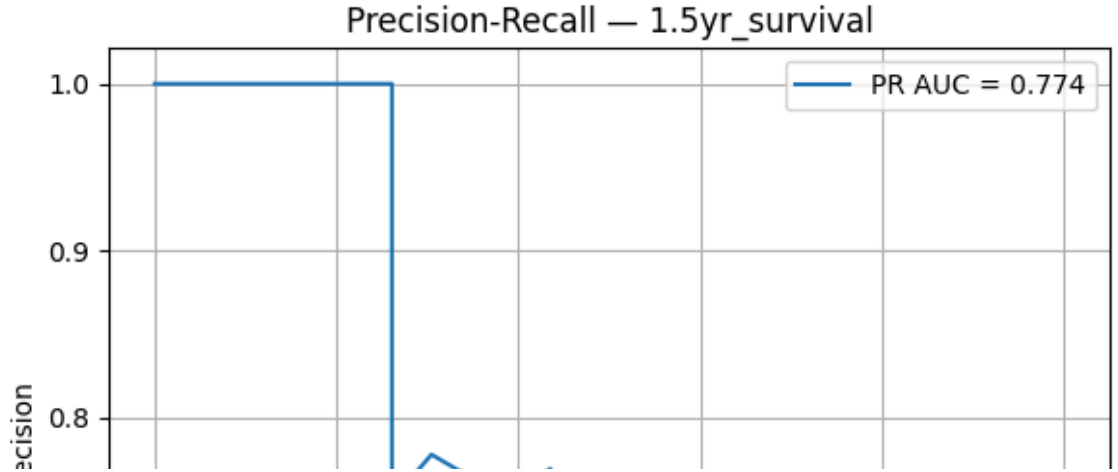
```
=== Training & Evaluating: 1.5yr_survival ===
Fitting 5 folds for each of 36 candidates, totalling 180 fits
  → XGB: best_params={'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200, 'subsample': 1.0}, AUC=0.668
Fitting 5 folds for each of 4 candidates, totalling 20 fits
  → RF: best_params={'max_depth': 4, 'n_estimators': 200}, AUC=0.601
Fitting 5 folds for each of 6 candidates, totalling 30 fits
  → SVM: best_params={'svc__C': 10, 'svc__kernel': 'rbf'}, AUC=0.519
  → Stacking AUC = 0.693
```

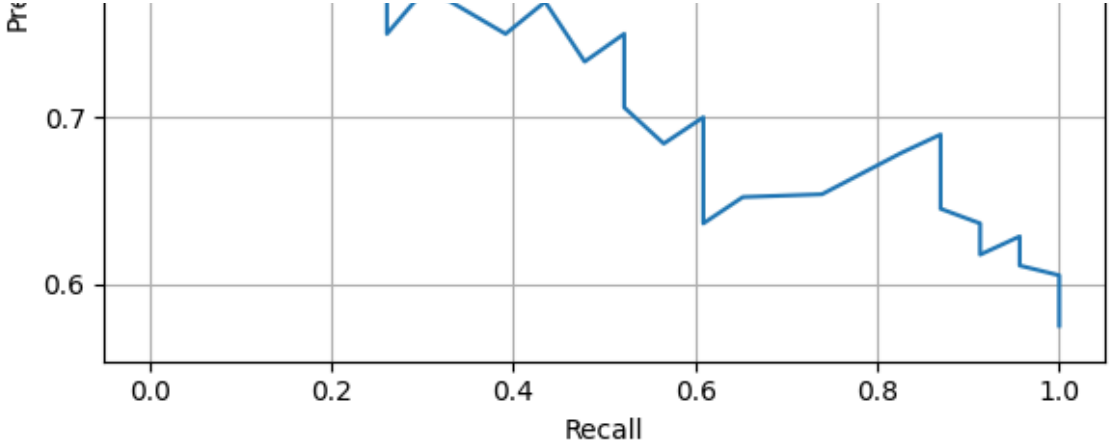


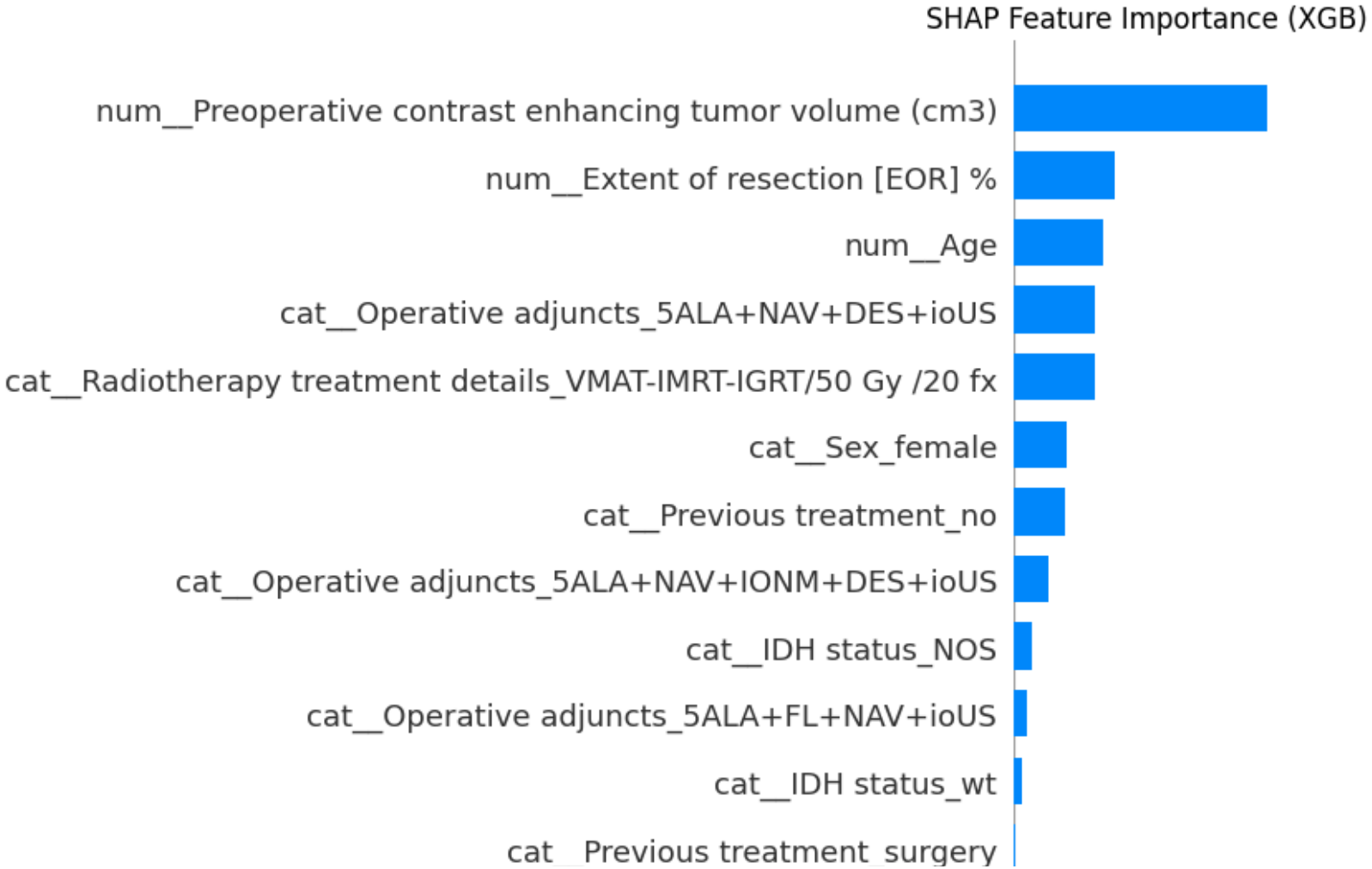
	precision	recall	f1-score	support
0	0.52	0.76	0.62	17
1	0.73	0.48	0.58	23
accuracy			0.60	40

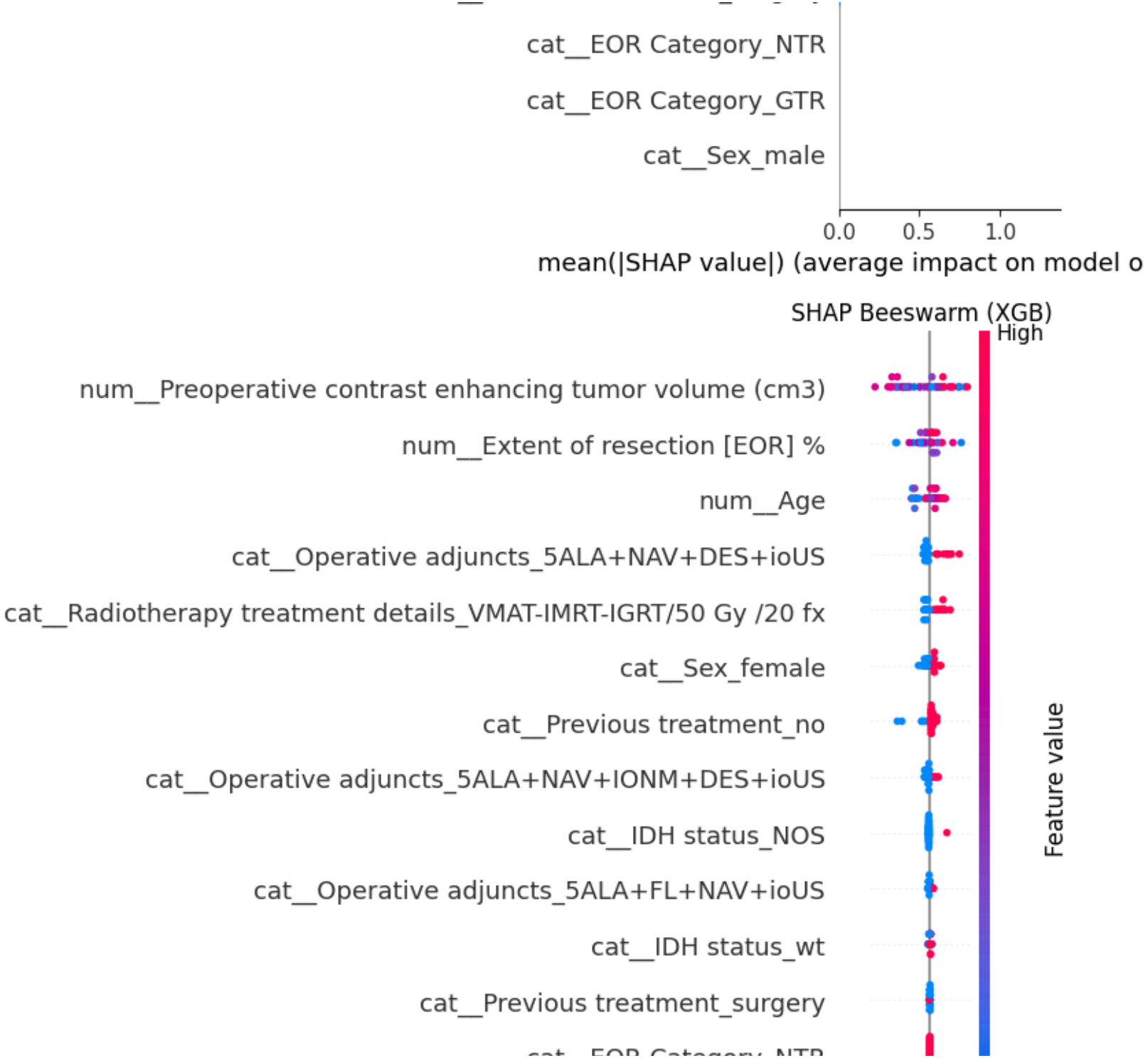
macro avg	0.63	0.62	0.60	40
weighted avg	0.64	0.60	0.60	40

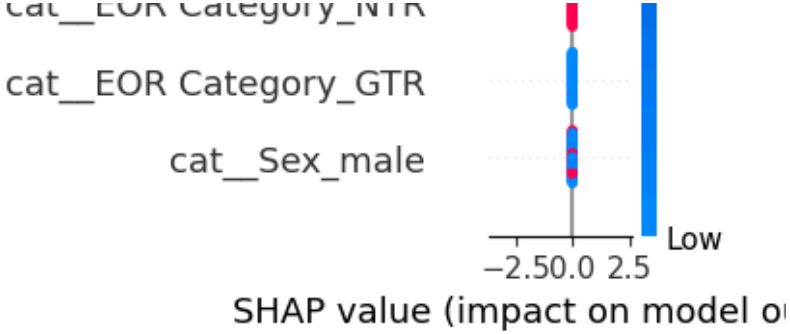




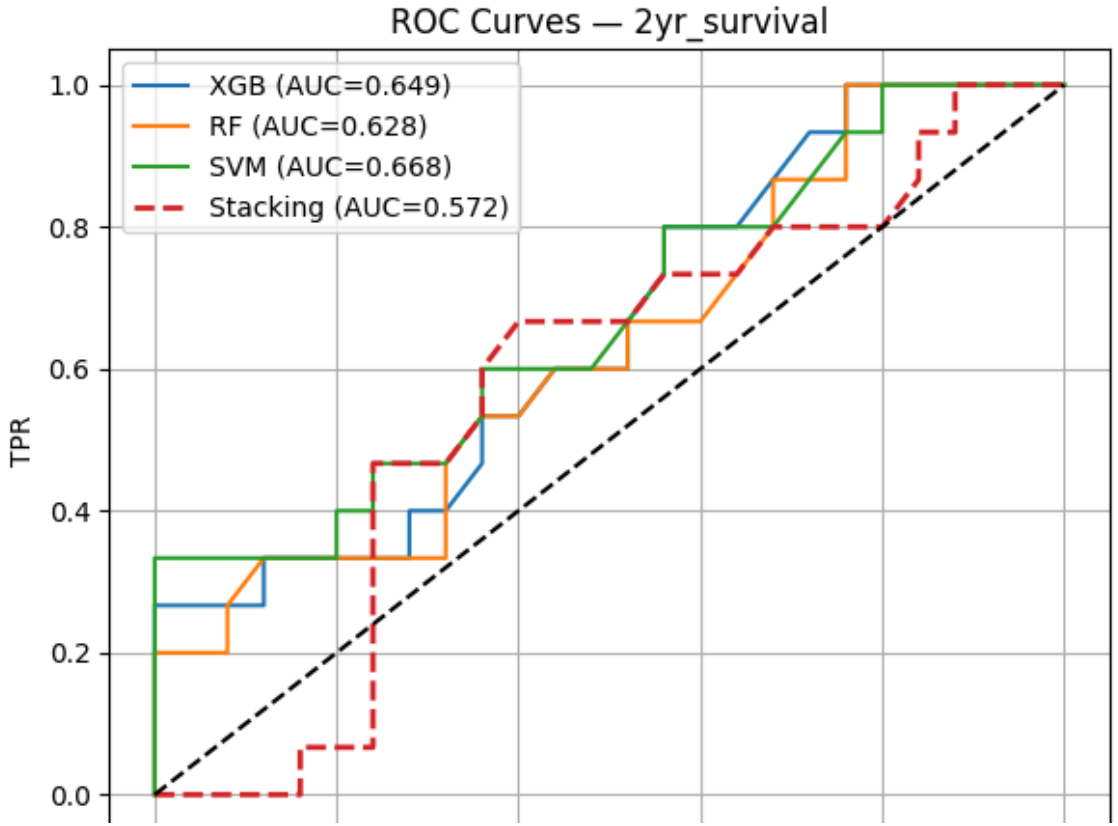

0.7
0.6
0.0
0.2
0.4
0.6
0.8
1.0
Recall


SHAP Feature Importance (XGB)
num__Preoperative contrast enhancing tumor volume (cm3)
num__Extent of resection [EOR] %
num__Age
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/50 Gy /20 fx
cat__Sex_female
cat__Previous treatment_no
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__IDH status_NOS
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__IDH status_wt
cat__Previous treatment_surgery

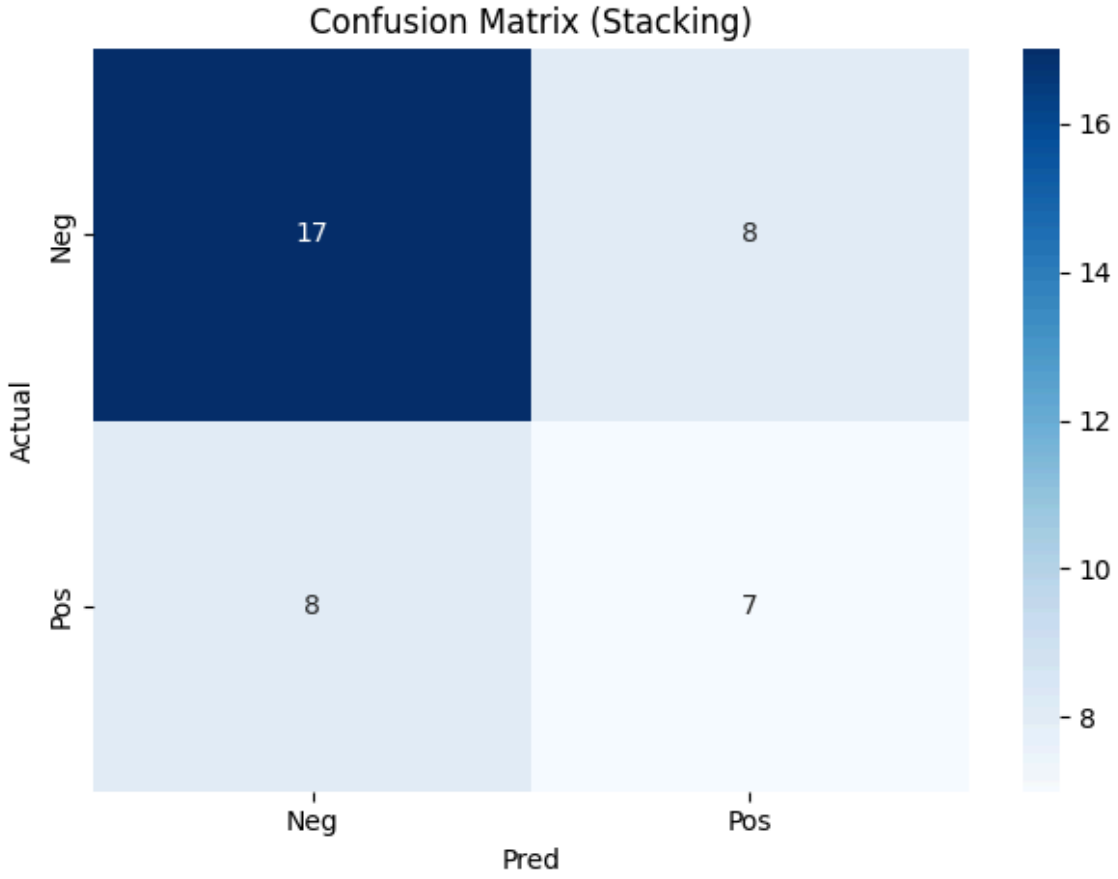

cat__EOR Category_NTR
cat__EOR Category_GTR
cat__Sex_male
0.0
0.5
1.0
mean(|SHAP value|) (average impact on model o
SHAP Beeswarm (XGB)
num__Preoperative contrast enhancing tumor volume (cm3)
num__Extent of resection [EOR] %
num__Age
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/50 Gy /20 fx
cat__Sex_female
cat__Previous treatment_no
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__IDH status_NOS
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__IDH status_wt
cat__Previous treatment_surgery
High
Feature value

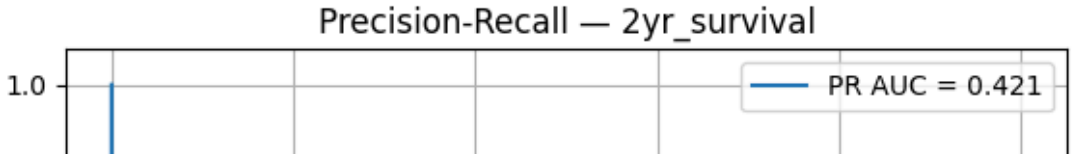


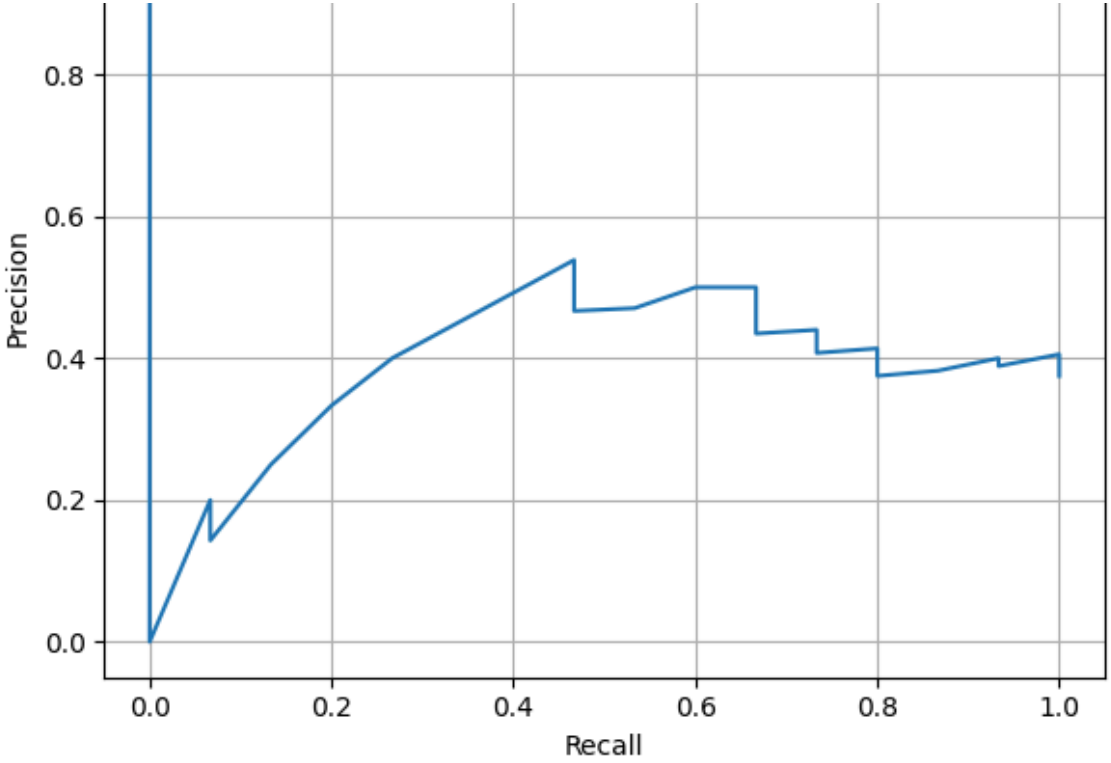

```
=== Training & Evaluating: 2yr_survival ===
Fitting 5 folds for each of 36 candidates, totalling 180 fits
  → XGB: best_params={'learning_rate': 0.05, 'max_depth': 4, 'n_estimators': 200, 'subsample': 0.8}, AUC=0.649
Fitting 5 folds for each of 4 candidates, totalling 20 fits
  → RF: best_params={'max_depth': 5, 'n_estimators': 100}, AUC=0.628
Fitting 5 folds for each of 6 candidates, totalling 30 fits
  → SVM: best_params={'svc__C': 10, 'svc__kernel': 'rbf'}, AUC=0.668
  → Stacking AUC = 0.572
```

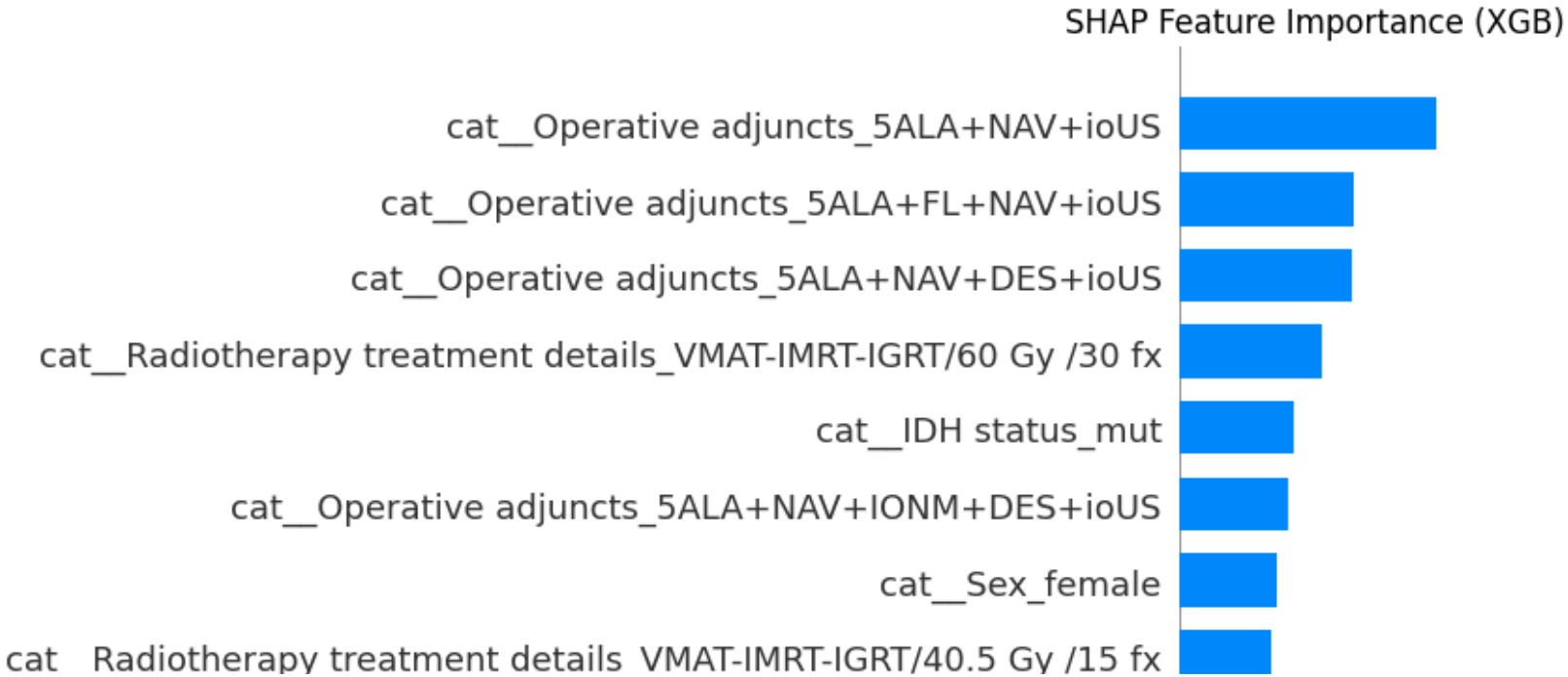


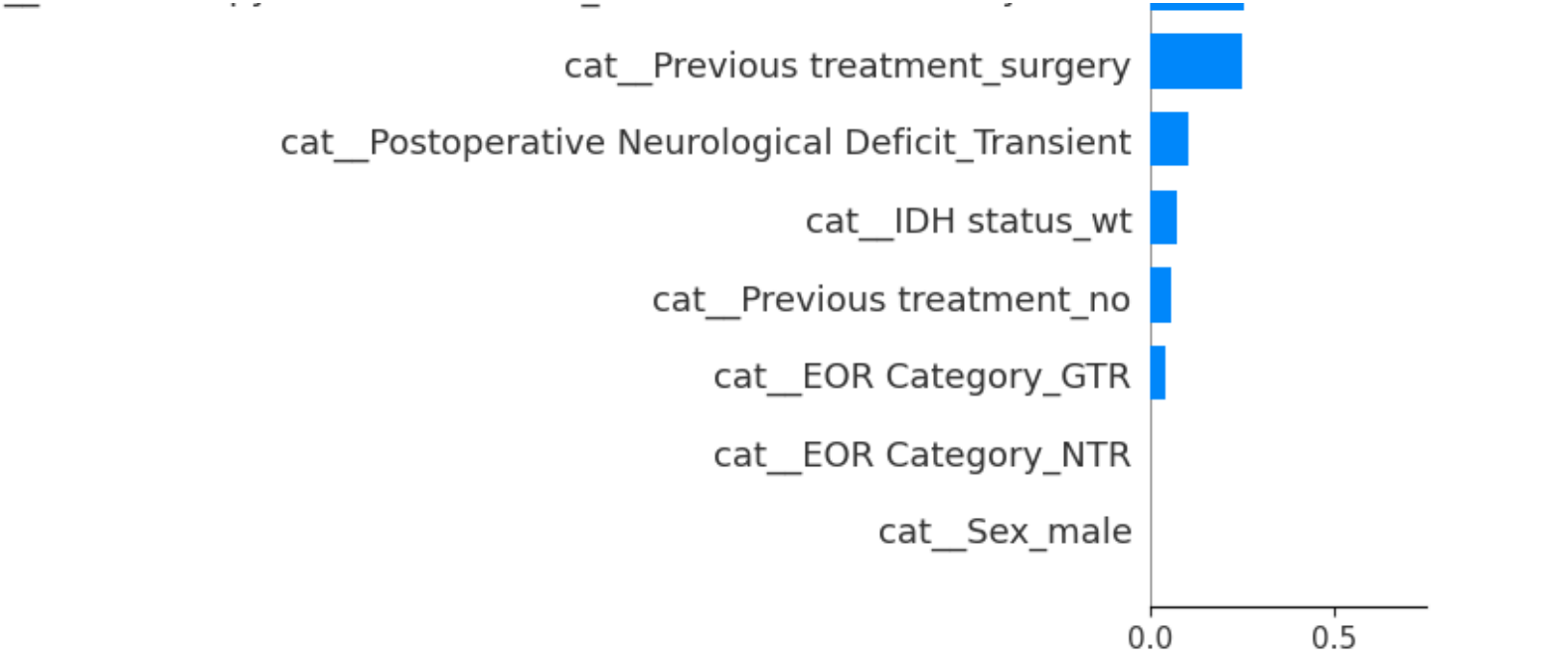
	precision	recall	f1-score	support
0	0.68	0.68	0.68	25
1	0.47	0.47	0.47	15
accuracy			0.60	40
macro avg	0.57	0.57	0.57	40
weighted avg	0.60	0.60	0.60	40

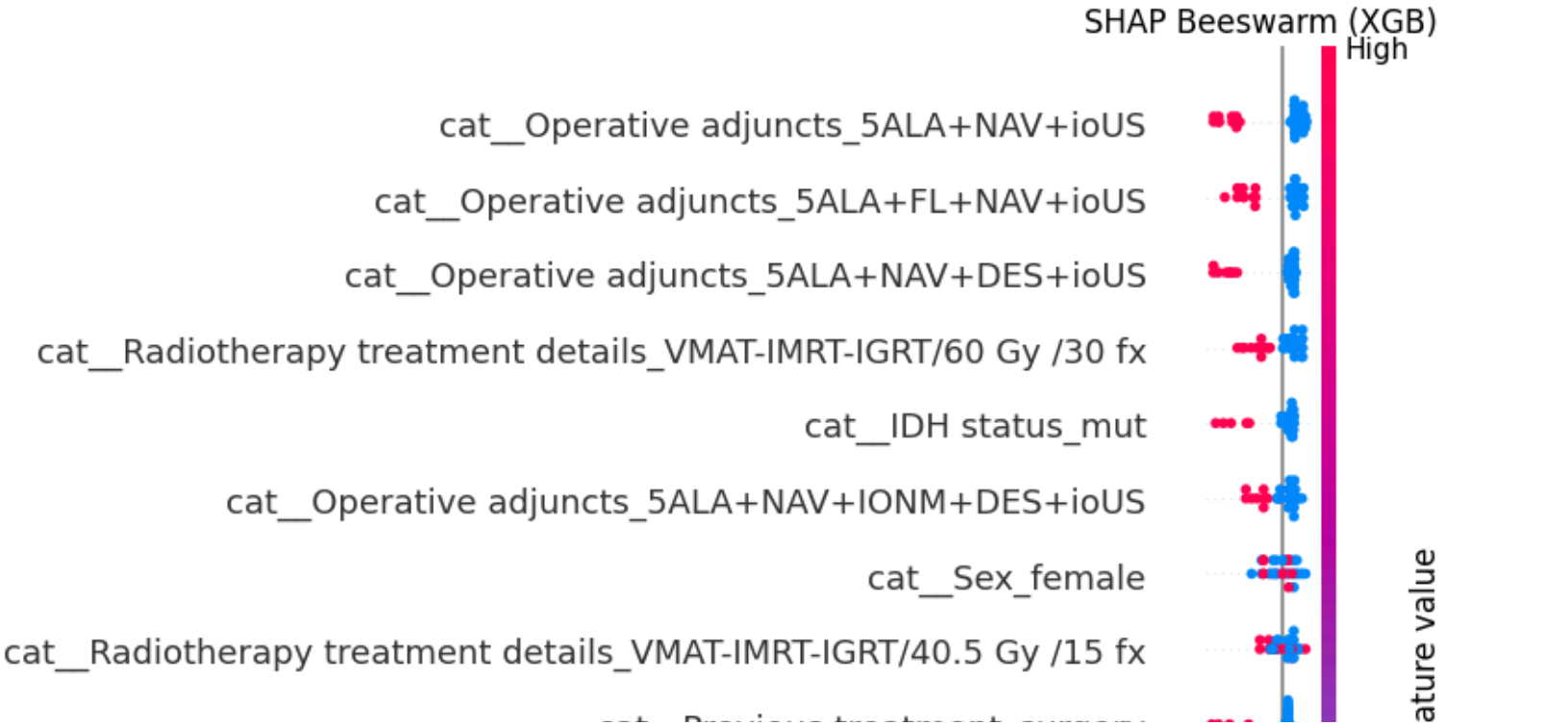


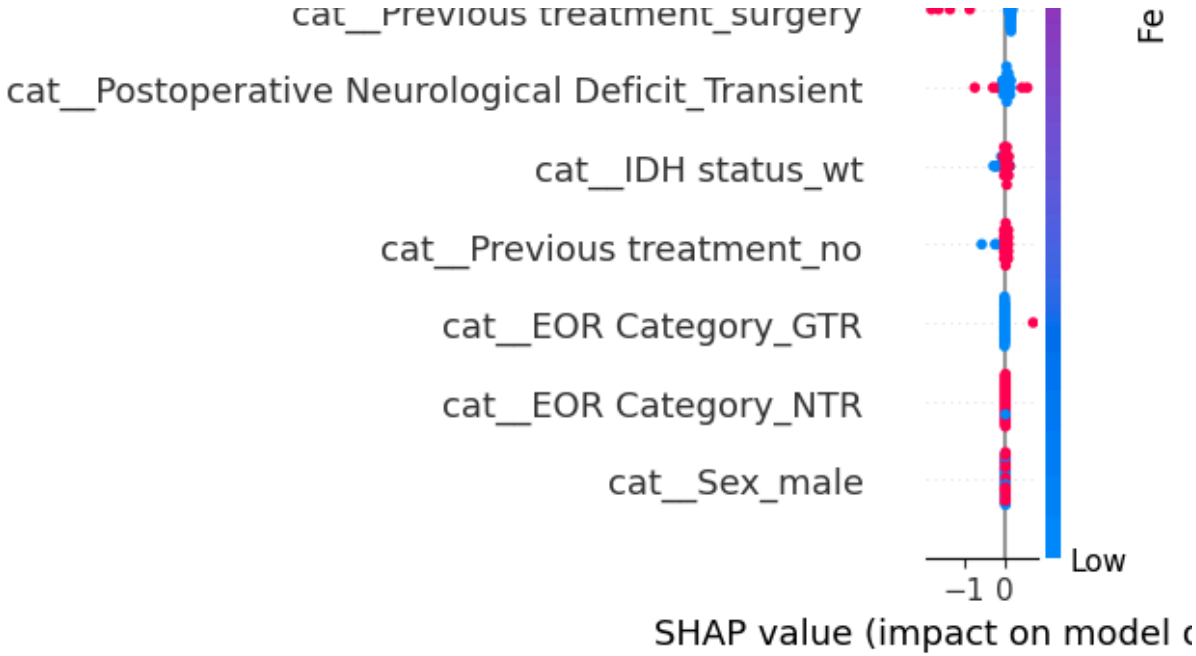



Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

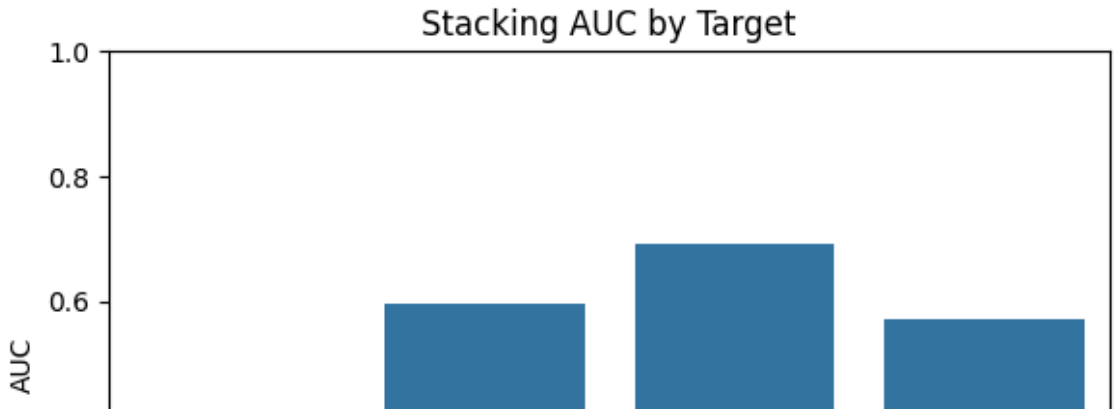

SHAP Feature Importance (XGB)
cat__Operative adjuncts_5ALA+NAV+ioUS
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/60 Gy /30 fx
cat__IDH status_mut
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__Sex_female
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/40.5 Gy /15 fx

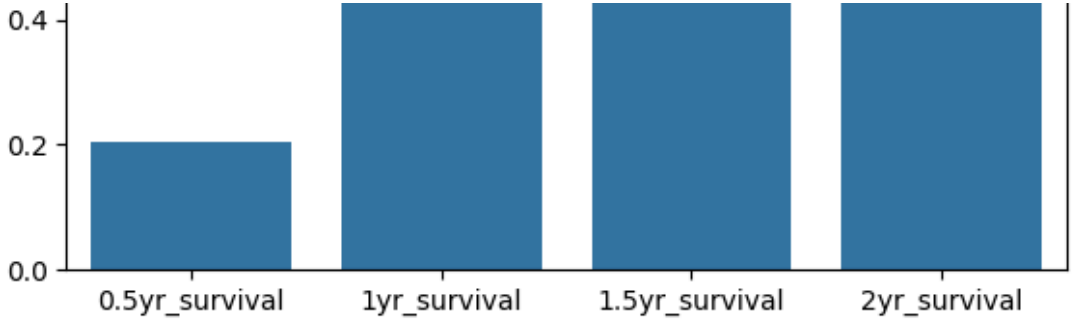

cat__Previous treatment_surgery
cat__Postoperative Neurological Deficit_Transient
cat__IDH status_wt
cat__Previous treatment_no
cat__EOR Category_GTR
cat__EOR Category_NTR
cat__Sex_male
0.0
0.5
mean(|SHAP value|) (average impact on model o

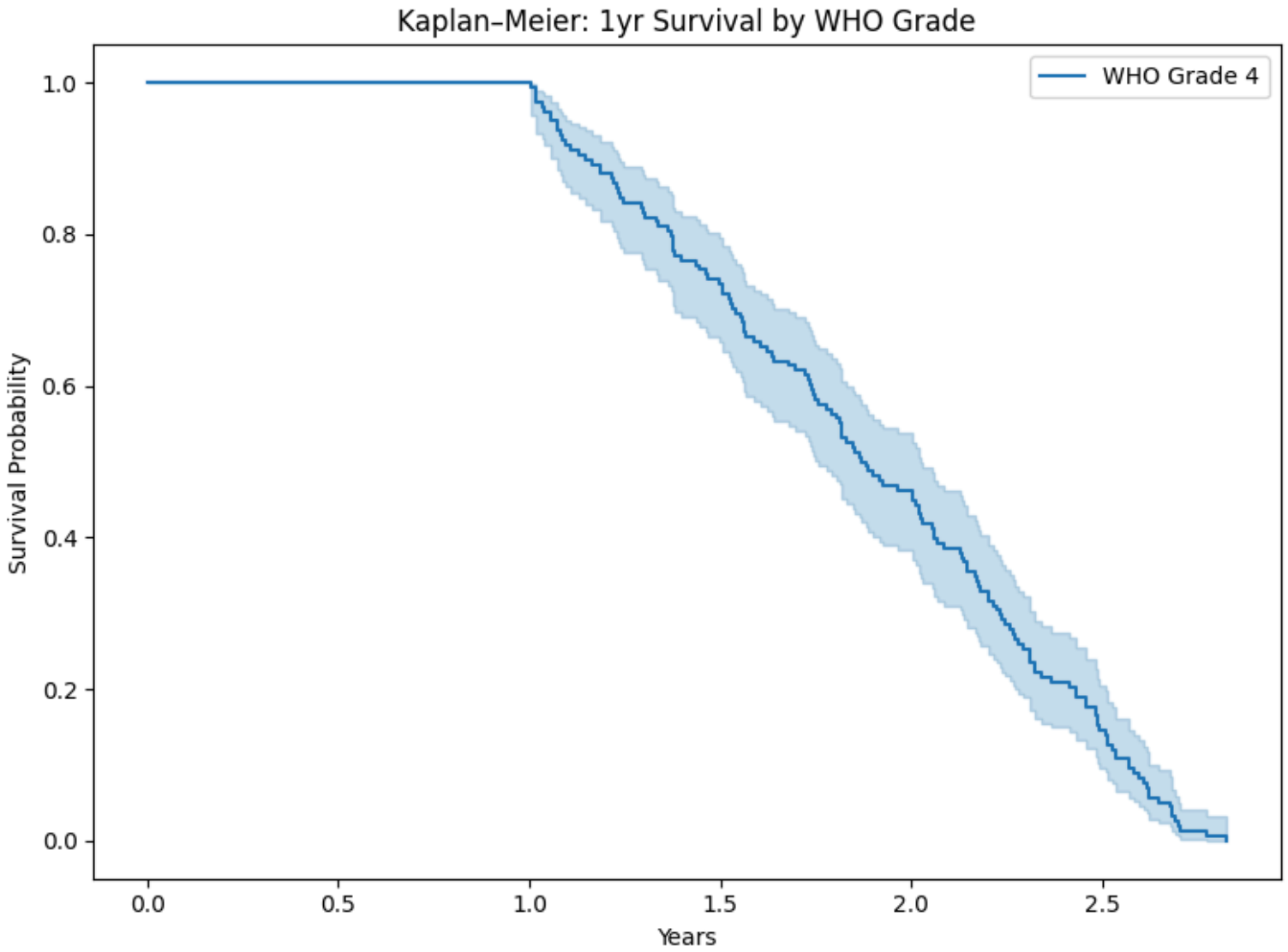

SHAP Beeswarm (XGB)
High
cat__Operative adjuncts_5ALA+NAV+ioUS
cat__Operative adjuncts_5ALA+FL+NAV+ioUS
cat__Operative adjuncts_5ALA+NAV+DES+ioUS
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/60 Gy /30 fx
cat__IDH status_mut
cat__Operative adjuncts_5ALA+NAV+IONM+DES+ioUS
cat__Sex_female
cat__Radiotherapy treatment details_VMAT-IMRT-IGRT/40.5 Gy /15 fx
ature value




```
=== Base Model AUCs by Target ===
0.5yr_survival: XGB=0.821, RF=0.949, SVM=0.718
1yr_survival: XGB=0.535, RF=0.621, SVM=0.586
1.5yr_survival: XGB=0.668, RF=0.601, SVM=0.519
2yr_survival: XGB=0.649, RF=0.628, SVM=0.668

=== Stacking AUCs ===
0.5yr_survival: 0.205
1yr_survival: 0.596
1.5yr_survival: 0.693
2yr_survival: 0.572
```




0.4
0.2
0.0
0.5yr_survival
1yr_survival
1.5yr_survival
2yr_survival


Kaplan–Meier: 1yr Survival by WHO Grade
WHO Grade 4
Survival Probability
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0
2.5
Years

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.