NET Early Detection AI Model

SAT5114 2025-Spring Project

Student: *Sankardu Varadapureddi*
Mentor: *Prof Dr. Guy Hembroff*

NET Early Detection AI Model - Overview

- NETs (Neuroendocrine Tumors) are rare and often undetected until advanced stages
- Lack of screening tools and symptom ambiguity contributes to poor prognosis
- **Goal:** Build an AI model that uses clinical data to identify early NET risk
- **Target prediction:** Overall Survival Status (OS_STATUS) – DECEASED or LIVING

Objectives and Long-Term Vision

Short Term Goals:

- Build classification models to predict survival outcomes
- Evaluate key features like SMOKER, CLINICAL_STAGE, etc.
- Handle imbalanced data by setting class_weight='balanced' in model training

Long Term Vision:

- Apply similar model to All of Us Controller Tier datasets
- Include genomics and NLP-extracted symptom data
- Integrate explainability for clinician use

Literature Review

- **Zhang, X., Liu, W., Li, Y., & Wang, Q. (2022).** *Predicting histologic grades for pancreatic neuroendocrine tumors by radiologic image based artificial intelligence: A systematic review and meta-analysis. Frontiers in Oncology, 14, Article 1332387*. (*https://www.frontiersin.org/journals/oncology/articles/10.3389/fonc.2024.1332387/full*):
  - Utilized random forest and gradient boosting on 200+ NET cases from different databases.
  - Reported AUC-ROC range of 0.83–0.89 depending on the subtype and tumor stage
  - Emphasized age, metastasis, tumor grade, and treatment history as top predictive features consistent with our findings (e.g., clinical stage, histology, SMOKER).

- **Song, Y., Zhang, J., Tian, Y., He, W., & Wang, G. (2018).** *Multiple machine learning models reveal similar predictive accuracy in pancreatic NETs. BMC Medical Informatics and Decision Making, 18(Suppl 5), 122,(https://pmc.ncbi.nlm.nih.gov/articles/PMC6218767/)* .
  - Compared ML models (LR, SVM, RF) and used oversampling to deal with class imbalance.
  - Used imputation strategies to handle missing values and addressed class imbalance during training

Dataset and Preprocessing

Datasets:

- panet_msk_2018 and panet_arcnet_2017 (https://cbioportal-datahub.s3.amazonaws.com/panet_msk_2018.tar.gz and https://cbioportal-datahub.s3.amazonaws.com/panet_arcnet_2017.tar.gz)
- Combined: 178 NET patient records

Preprocessing Workflow:

- **Feature Alignment:** Selected 14 shared columns (e.g., AGE, SEX, SMOKER) and filled missing ones with NaN for consistent merging.
- **Label Encoding:** Converted categorical columns (e.g., SEX, ETHNICITY) to numeric using LabelEncoder.
- **Missing Data Handling:** Used SimpleImputer(strategy='most_frequent') to fill missing values in both numeric and categorical fields.

Dataset and Preprocessing..contd

Preprocessing Workflow:

- **Outcome Definition:** Encoded OS_STATUS as 1 (DECEASED) and 0 (LIVING) for binary classification.
- **Train-Test Split:** Split data into 80% train and 20% test using train_test_split(random_state=42).
- **Class Imbalance:** Set class_weight='balanced' in models to compensate for survival class imbalance.
- **Feature Scaling:** Applied StandardScaler() within pipelines for SVM and Logistic Regression to normalize inputs.

```
df_merged['OS_STATUS'] = df_merged['OS_STATUS'].apply(lambda x: 1 if 'DECEASED' in str(x) else 0)

imputer = SimpleImputer(strategy='most_frequent')

X_train_resampled = SMOTE().fit_resample(X_train, y_train)
```

Machine Learning Models

Models Evaluated :

- Logistic Regression
- Random Forest
- SVM with RBF Kernel
- K-Nearest Neighbors (KNN)
- Naïve Bayes

Why these?

- Covers linear, tree-based, kernel-based, distance-based, and probabilistic paradigms
- Provides comparative insight on performance and interpretability

```
models = {
    "Logistic Regression": make_pipeline(StandardScaler(), LogisticRegression(max_iter=1000, class_weight='balanced')),
    "Random Forest": RandomForestClassifier(n_estimators=100, class_weight='balanced', random_state=42),
    "SVM (RBF Kernel)": make_pipeline(StandardScaler(), SVC(kernel='rbf', probability=True, class_weight='balanced')),
    "K-Nearest Neighbors": KNeighborsClassifier(n_neighbors=5),
    "Naive Bayes": GaussianNB()
}
```

Evaluation Metrics

Key Evaluation Metrics :

- **Accuracy**: Overall correctness of predictions
- **Precision**: Of predicted positives, how many were correct
- **Recall**: Of actual positives, how many were detected
- **F1-Score**: Balance between precision and recall
- **AUC-ROC**: Overall ability to distinguish classes

Why these?

- In clinical AI, **recall and AUC** are often prioritized over plain accuracy
- **High recall** ensures most at-risk patients are flagged, minimizing missed NET-positive cases
- **AUC-ROC** ensures the model is capable of distinguishing between risk levels even at different thresholds

Feature Interpretation

Random Forest Top Features:

Feature	Importance
OS_MONTHS	Highest
AGE	High
CLINICAL_STAGE	Moderate
SURGICAL_MARGIN	Moderate
SMOKER, SEX, DRINKER	Notable

```
rf_model = model_objects["Random Forest"][0]
importances = rf_model.feature_importances_
```

Statistical Significance Test (Chi-Square):

- SMOKER vs OS_STATUS → Significant ($p < 0.05$)
- DRINKER → Not significant

```
print("\\n=== Chi-Square Test: Lifestyle vs Outcome ===")
for feature in ['SMOKER', 'DRINKER']:
    table = pd.crosstab(df_merged_imputed[feature], df_merged_imputed['OS_STATUS'])
    chi2, p, _, _ = chi2_contingency(table)
    print(f"{feature}: p-value = {p:.4f} {'(Significant)' if p < 0.05 else '(Not significant)'}")
```

Performance Summary

✅ **Random Forest** achieves the best **AUC-ROC (0.92)** — excellent at distinguishing outcomes, but overfits slightly (perfect train scores). (Note: since recall was low, especially for the DECEASED class — this is being further investigated in future work)

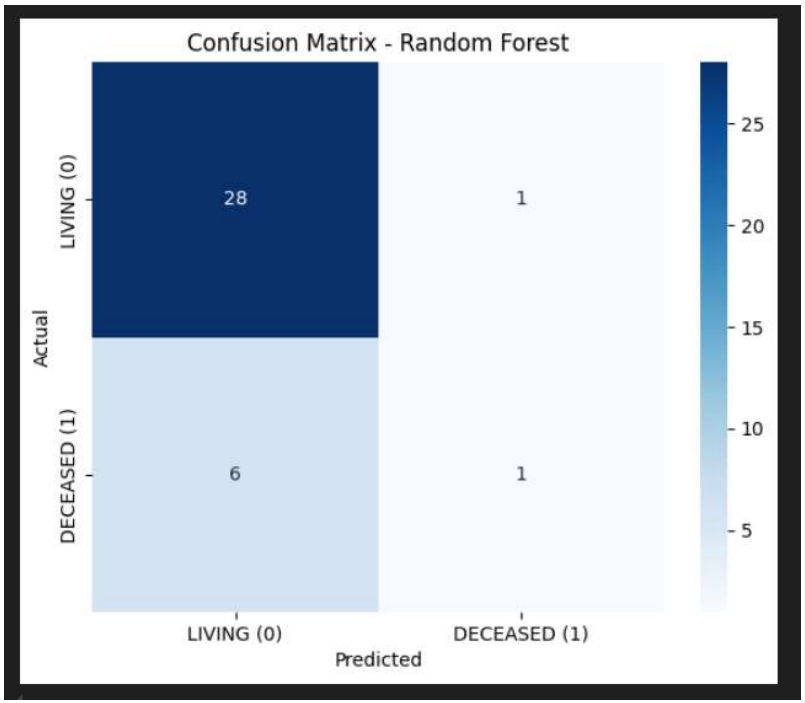
✅ **Logistic Regression** has the **highest recall (0.85)** on test set — ideal for **medical screening**, where false negatives must be minimized

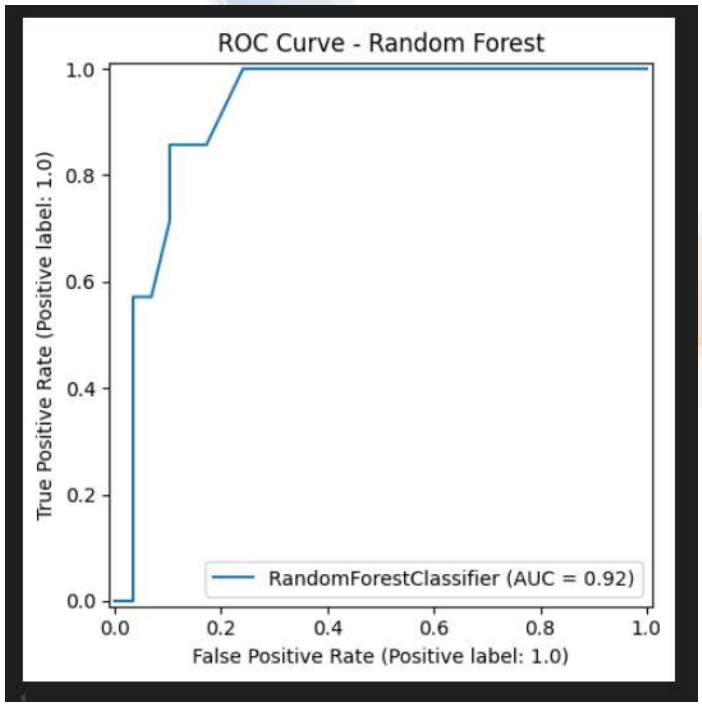
🔄 **Naïve Bayes** and **KNN** perform moderately well; KNN collapses on unseen test data (zero recall)

❌ **SVM** shows high variance: excellent training metrics but generalizes poorly — possibly due to RBF sensitivity on limited samples

```
=== Training Set Performance ===

                Model  Accuracy  Precision    Recall  F1-Score   AUC-ROC
1       Random Forest  1.000000   1.000000  1.000000  1.000000  1.000000
0 Logistic Regression  0.718310   0.333333  0.818182  0.473684  0.859848
3 K-Nearest Neighbors  0.852113   0.666667  0.090909  0.160000  0.842803
2           SVM (RBF)  0.626761   0.293333  1.000000  0.453608  0.840909
4         Naive Bayes  0.816901   0.423077  0.500000  0.458333  0.775758

=== Test Set Performance ===

                Model  Accuracy  Precision    Recall  F1-Score   AUC-ROC
1       Random Forest  0.805556   0.500000  0.142857  0.222222  0.923645
4         Naive Bayes  0.750000   0.416667  0.714286  0.526316  0.748768
0 Logistic Regression  0.611111   0.315789  0.857143  0.461538  0.724138
3 K-Nearest Neighbors  0.777778   0.000000  0.000000  0.000000  0.719212
2           SVM (RBF)  0.472222   0.250000  0.857143  0.387097  0.605911
```

Visual Insights – Model Evaluation (Random Forest)





- High **specificity** (correctly classifies LIVING cases)
- Low **sensitivity (recall)** – 6 out of 7 DECEASED cases were missed
- Serious concern in **medical applications** where missing a sick patient is risky

- AUC of **0.92** indicates **excellent ability** to distinguish between classes
- Suggests model knows the pattern but is conservative in predicting DECEASED

Challenges Faced

Data Access:

- Couldn't use All of Us directly due to Controlled Tier access and restrictions
- Relied on public NET datasets

Data Quality Issues:

- Missing values handled with SimpleImputer
- Feature alignment across datasets

Class Imbalance:

- NET-positive cases underrepresented
- Addressed by setting class_weight='balanced' in model definitions (e.g., Logistic Regression, Random Forest)

Python Learning Curve:

- Gained new experience with pipelines, imbalanced-learn

Future Work

- Investigate threshold tuning or SMOTE in future iterations to improve recall on underrepresented classes
- Migrate model to All of Us Controlled Tier dataset
- Include EHR + genomic + NLP-based symptom data
- Implement SHAP for explainability
- Work with real clinical datasets and validation in hospital systems

Thank you