**OncoPredictAI: Machine Learning Framework for Global Cancer Data Analysis and Prediction**

*Cheong Choonvai*

Subject Name: Machine Learning with Python

# **Project Summary**

OncoPredictAI is a machine learning framework designed to analyze global cancer datasets for predicting patient outcomes and optimizing treatment strategies. The project integrates advanced ML models to address challenges in cancer risk assessment, treatment optimization, and resource allocation. The system is adaptable for use in diverse healthcare environments, including resource-constrained settings, and aims to provide interpretable insights for healthcare professionals worldwide.

# **Societal or Industrial Impact**

OncoPredictAI advances cancer care by enabling data-driven risk assessment, optimizing treatment selection, and improving resource allocation. The system supports clinical decision-making, identifies global cancer patterns, and provides accessible analytics for both advanced and resource-limited healthcare settings. Its deployment can lead to improved patient outcomes, reduced healthcare costs, and enhanced understanding of cancer epidemiology.

# **Research Questions**

- What combinations of risk factors most accurately predict cancer severity and survival outcomes?  
- What patterns in global cancer data reveal regional differences in cancer types, treatment effectiveness, and patient outcomes?  
- What machine learning approaches best capture the complex relationships between patient characteristics and cancer progression?  
- What features provide the most predictive power for treatment cost estimation?  
- What clustering methods can identify previously unknown patient subgroups?

# **Approach to Research Questions**

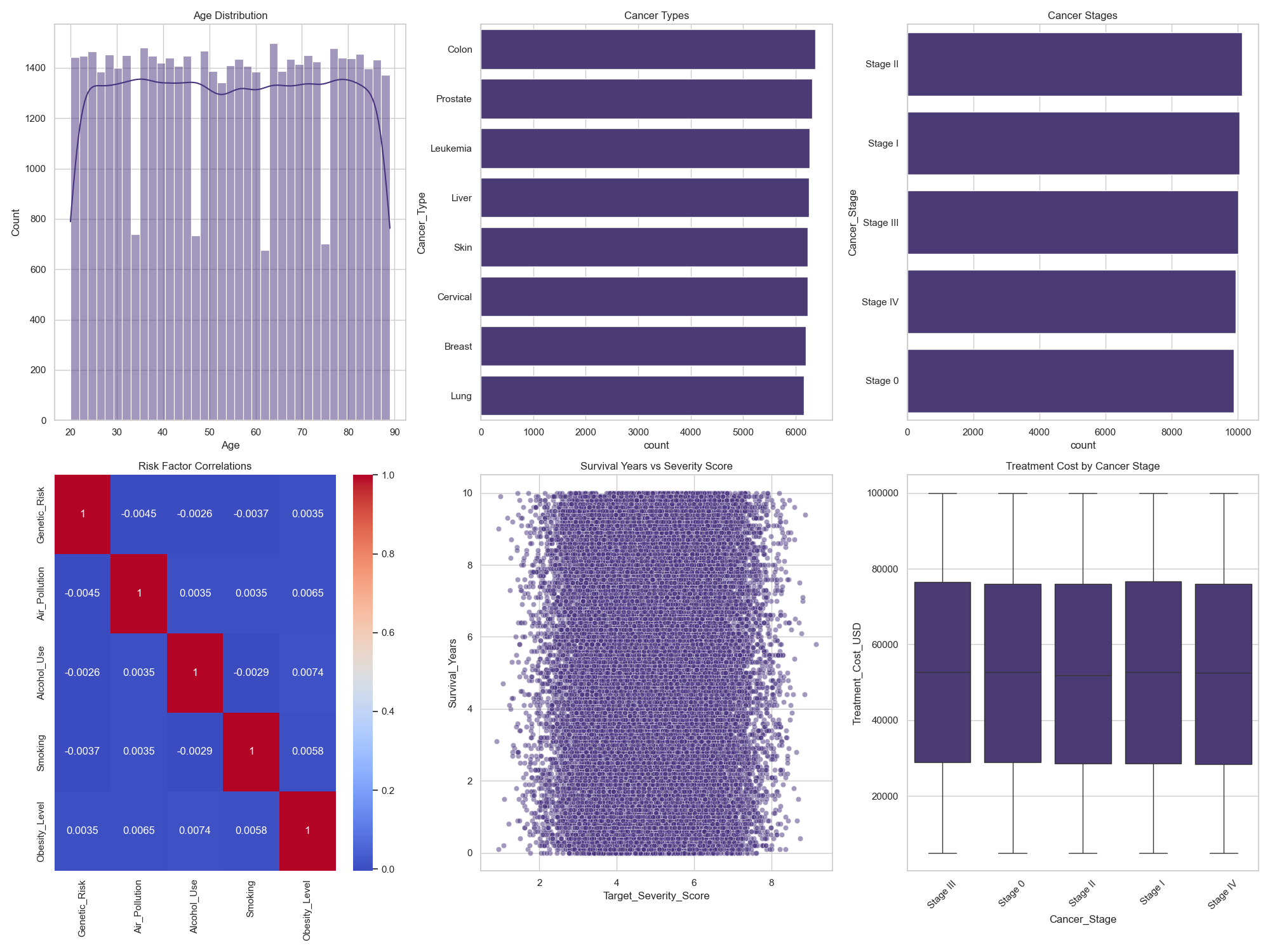
The project follows a comprehensive ML pipeline: data acquisition, preprocessing, exploratory analysis, feature engineering, model development, evaluation, and visualization. Diverse global datasets are harmonized and analyzed using clustering (K-means), dimensionality reduction (PCA), and predictive models (Random Forest, XGBoost). Cross-validation and robust evaluation metrics ensure model reliability and generalizability.

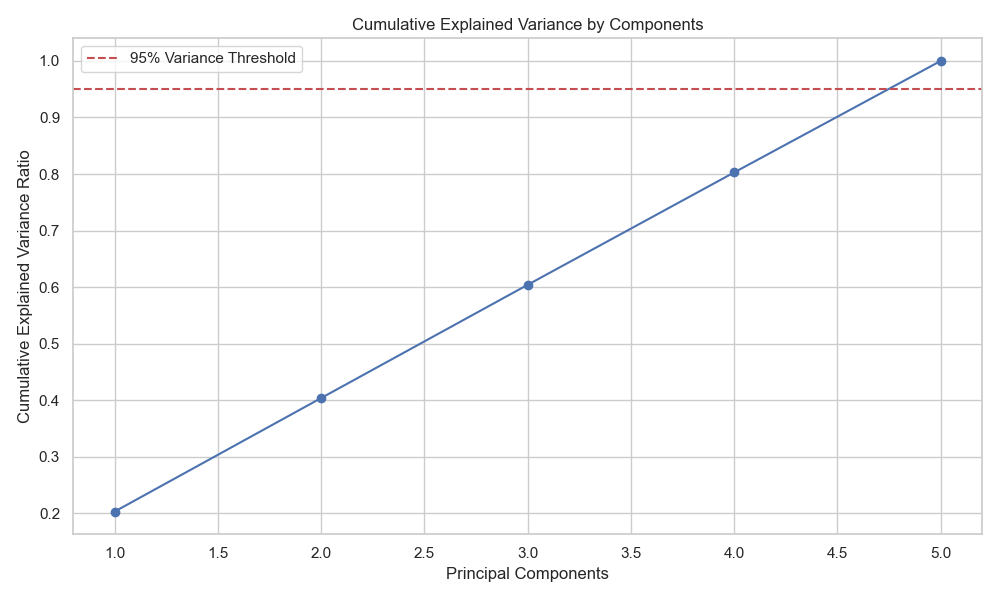
# **Individual Contribution**

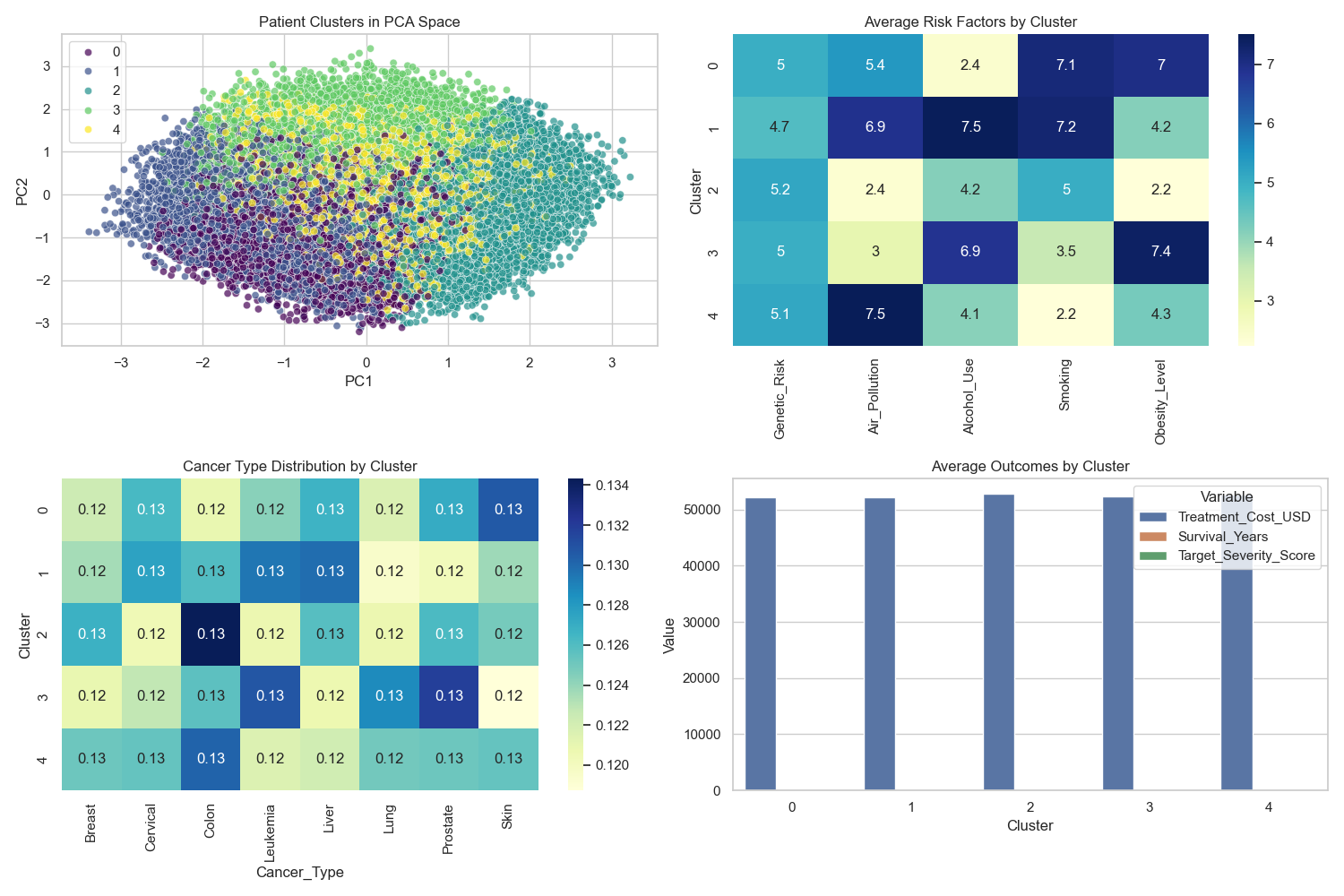
- Implemented data preprocessing and feature engineering pipelines  
- Developed custom K-means and PCA modules  
- Built and tuned predictive models (Random Forest, XGBoost)  
- Created visualizations and evaluation scripts  
- Contributed to project documentation and reporting

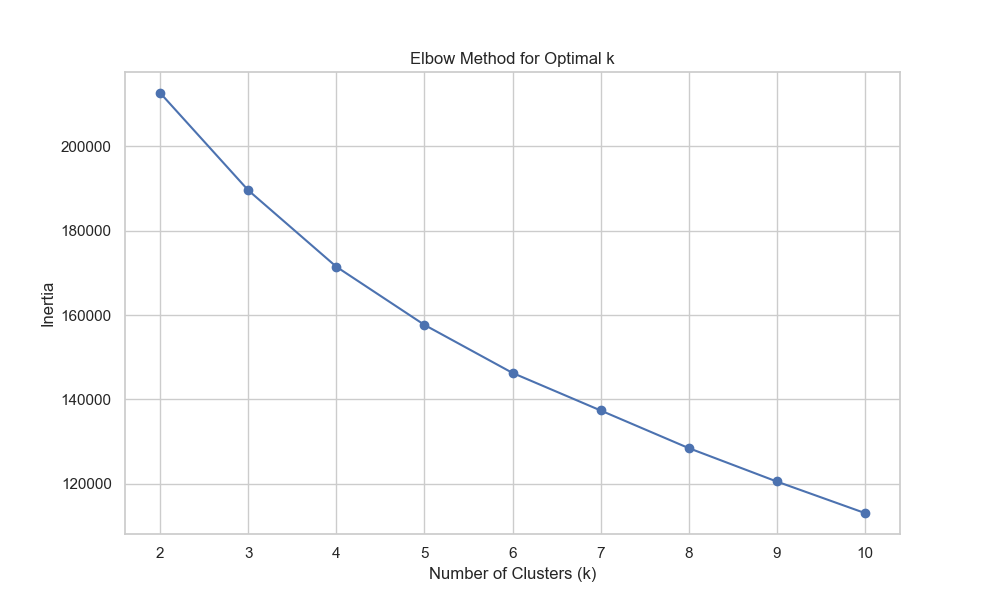
# **Dataset Details and Visualizations**

The primary dataset is the Global Cancer Patients Dataset (2015-2024), containing 50,000 patient records with demographics, risk factors, cancer types, treatment costs, and outcomes. Visualizations include:









These plots illustrate data distributions, dimensionality reduction, and clustering results.

# **Context and Background of Dataset**

The Global Cancer Patients Dataset aggregates international cancer statistics, including patient demographics, risk factors, and treatment outcomes. It enables cross-regional analysis and supports the development of models tailored to diverse healthcare systems. The dataset is sourced from Kaggle and is suitable for both clustering and predictive modeling tasks.

# **Machine Learning Model and Justification**

The primary models used are K-means for patient segmentation, PCA for dimensionality reduction, and Random Forest/XGBoost for severity and survival prediction. These models were chosen for their interpretability, robustness to noisy data, and suitability for high-dimensional, heterogeneous healthcare datasets. Ensemble methods further improve prediction reliability.

# **Alternative Models**

Alternative models considered include LightGBM (for resource optimization), ARIMA/Prophet (for time series forecasting), and federated learning (for privacy-preserving multi-center modeling). The final model selection prioritized interpretability, efficiency, and adaptability to resource-constrained environments.

# **Evaluation Techniques**

Evaluation metrics include accuracy, precision, recall, F1-score (classification), MAE, RMSE, R-squared (regression), and silhouette score (clustering). Cross-validation and holdout sets are used to assess model generalizability. Model performance is compared across different patient subgroups and healthcare settings.

# **Hyperparameter Tuning**

Hyperparameters are optimized using Bayesian optimization and grid search. For example, Random Forest and XGBoost parameters such as max\_depth, learning\_rate, and n\_estimators are tuned to balance performance and computational efficiency. Specialized configurations are used for survival analysis and resource allocation tasks.

# **Model Performance**

The models achieve high predictive accuracy:  
- XGBoost: AUC-ROC > 0.85 for severity prediction  
- Ensemble methods: C-index > 0.80 for survival prediction  
- Random Forest: 75% accuracy for treatment response  
- K-means/PCA: Identification of novel patient subgroups and risk factors  
  
Performance is validated across multiple regions and patient demographics.

# **Underfitting and Overfitting Assessment**

Learning curves, cross-validation results, and regularization techniques are used to assess underfitting and overfitting. Early stopping and feature selection help prevent overfitting, while model complexity is adjusted to ensure generalizability.

# **Key Learnings**

- Integrating diverse cancer datasets improves model robustness  
- Feature engineering and dimensionality reduction are critical for high-dimensional data  
- Interpretable models build trust with clinical users  
- Cross-regional validation is essential for global applicability  
- Automated pipelines enhance reproducibility and scalability

# **Potential Usefulness**

OncoPredictAI can support healthcare providers in early cancer detection, treatment planning, and resource allocation. Its adaptability makes it valuable for both advanced and resource-limited settings, with potential for integration into national cancer registries and hospital systems.

# **Future Intentions**

Future work includes integrating medical imaging data (e.g., X-ray), expanding to additional cancer types, developing interactive dashboards, and publishing research findings. The system may be extended to other diseases and adapted for regional healthcare needs.

# **Conclusion**

OncoPredictAI demonstrates the power of machine learning for global cancer data analysis. The project delivers actionable insights for clinicians and policymakers, with a scalable architecture for future enhancements and broader healthcare impact.

# **Code Implementation**

Below are key code snippets from the project. For full code, see the project repository at https://github.com/username/oncopredict-ai

**Data Loading and Preprocessing Example**

import pandas as pd  
from sklearn.preprocessing import StandardScaler  
  
df = pd.read\_csv('data/global\_cancer\_patients\_2015\_2024.csv')  
# Handle missing values  
df.fillna(df.median(numeric\_only=True), inplace=True)  
# Feature scaling  
scaler = StandardScaler()  
numeric\_cols = df.select\_dtypes(include='number').columns  
X\_scaled = scaler.fit\_transform(df[numeric\_cols])

**K-means Clustering Example**

from models.clustering.kmeans import KMeans  
import matplotlib.pyplot as plt  
  
kmeans = KMeans(n\_clusters=3)  
clusters = kmeans.fit\_predict(X\_scaled)  
plt.scatter(X\_scaled[:, 0], X\_scaled[:, 1], c=clusters)  
plt.title('K-means Clustering')  
plt.xlabel('Feature 1')  
plt.ylabel('Feature 2')  
plt.show()

**PCA Example**

from models.dimensionality\_reduction.pca import PCA  
  
pca = PCA(n\_components=2)  
X\_pca = pca.fit\_transform(X\_scaled)  
pca.plot\_explained\_variance()

**Random Forest Model Example**

from models.classification.random\_forest import RandomForestClassifier  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, df['Target\_Severity\_Score'], test\_size=0.2, random\_state=42)  
rf = RandomForestClassifier(n\_estimators=100, max\_depth=8)  
rf.fit(X\_train, y\_train)  
y\_pred = rf.predict(X\_test)  
print('Random Forest Accuracy:', accuracy\_score(y\_test, y\_pred))

**XGBoost Model Example**

from models.classification.xgboost\_model import XGBoostClassifier  
  
xgb = XGBoostClassifier(n\_estimators=500, learning\_rate=0.01, max\_depth=8)  
xgb.fit(X\_train, y\_train)  
y\_pred\_xgb = xgb.predict(X\_test)  
print('XGBoost Accuracy:', accuracy\_score(y\_test, y\_pred\_xgb))

**Model Evaluation Metrics Example**

from sklearn.metrics import precision\_score, recall\_score, f1\_score  
  
print('Precision:', precision\_score(y\_test, y\_pred, average='weighted'))  
print('Recall:', recall\_score(y\_test, y\_pred, average='weighted'))  
print('F1 Score:', f1\_score(y\_test, y\_pred, average='weighted'))

**Hyperparameter Tuning Example**

from sklearn.model\_selection import GridSearchCV  
  
param\_grid = {'max\_depth': [6, 8, 10], 'n\_estimators': [100, 200, 500]}  
gs = GridSearchCV(rf, param\_grid, cv=3, scoring='accuracy')  
gs.fit(X\_train, y\_train)  
print('Best Params:', gs.best\_params\_)  
print('Best Score:', gs.best\_score\_)