# Import necessary libraries

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.datasets import load\_iris

# Load the Iris dataset as an example

data = load\_iris()

X, y = data.data, data.target

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Base models

base\_model1 = RandomForestClassifier(n\_estimators=100, random\_state=42)

base\_model2 = GradientBoostingClassifier(n\_estimators=100, random\_state=42)

# Meta-model

meta\_model = LogisticRegression()

# Train the base models

base\_model1.fit(X\_train, y\_train)

base\_model2.fit(X\_train, y\_train)

# Generate predictions from the base models

pred1 = base\_model1.predict(X\_train)

pred2 = base\_model2.predict(X\_train)

from sklearn.metrics import accuracy\_score

accuracy = accuracy\_score(y\_train, pred1)

print(f"Accuracy of the base model RF: {accuracy}")

from sklearn.metrics import accuracy\_score

accuracy = accuracy\_score(y\_train, pred2)

print(f"Accuracy of the Base model GB: {accuracy}")

# Create a new feature matrix with base model predictions

X\_stacked = np.array([pred1, pred2]).T

print(X\_stacked)

# Train the meta-model on the base model predictions

meta\_model.fit(X\_stacked, y\_train)

# Now, for making predictions on new data, first get predictions from the base models

base\_pred1 = base\_model1.predict(X\_test)

base\_pred2 = base\_model2.predict(X\_test)

# Create a feature matrix for the test data using base model predictions

X\_test\_stacked = np.array([base\_pred1, base\_pred2]).T

# Make final predictions using the meta-model

final\_predictions = meta\_model.predict(X\_test\_stacked)

# Evaluate the final model

from sklearn.metrics import accuracy\_score

accuracy = accuracy\_score(y\_test, final\_predictions)

print(f"Accuracy of the stacked model: {accuracy}"