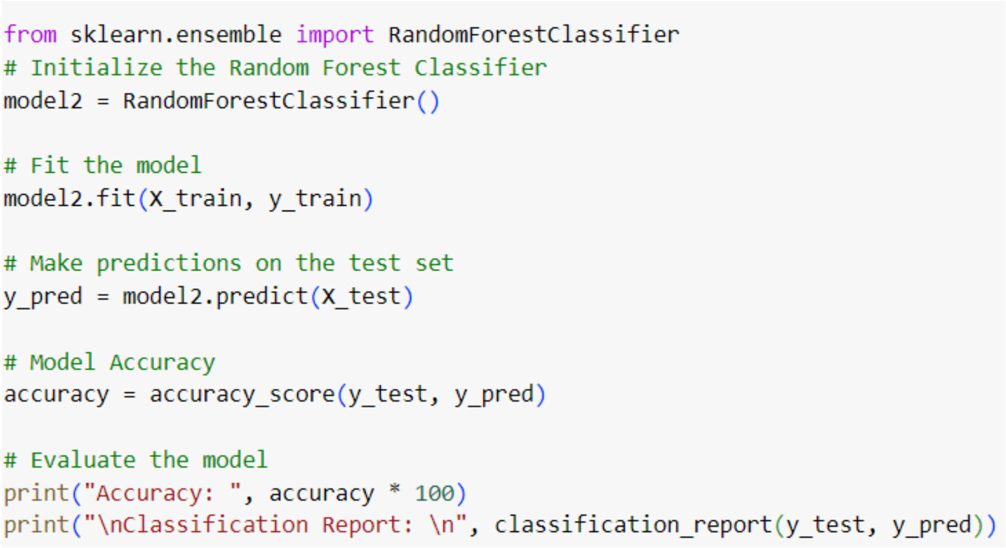
Model Development Phase Template

Date	21 June 2024
Team ID	739769
Project Title	Life Style Change Due To Covid Prediction
Maximum Marks	4 Marks

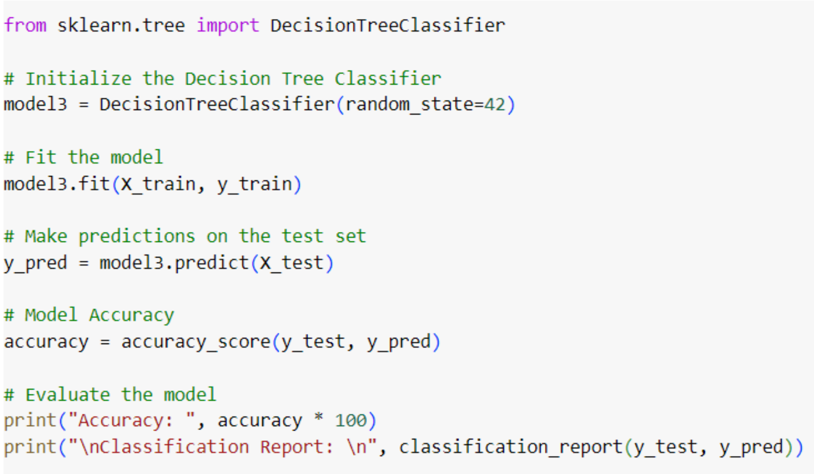
Initial Model Training Code, Model Validation and Evaluation Report

The initial Random Forest model shows promising results for predicting lifestyle changes due to COVID-19 based on demographic and behavioral attributes. Further refinement of the model, including hyperparameter tuning and feature engineering, may enhance its predictive performance.

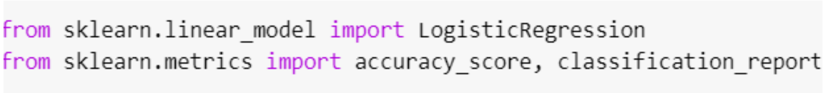
Initial Model Training Code:



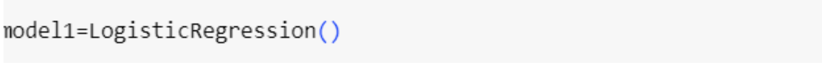
```
from sklearn.ensemble import RandomForestClassifier
# Initialize the Random Forest Classifier
model2 = RandomForestClassifier()

# Fit the model
model2.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model2.predict(X_test)

# Model Accuracy
accuracy = accuracy_score(y_test, y_pred)

# Evaluate the model
print("Accuracy: ", accuracy * 100)
print("\nClassification Report: \n", classification_report(y_test, y_pred))
```



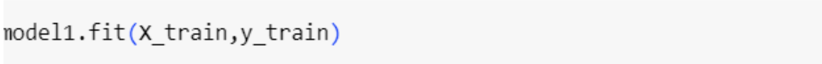
```
from sklearn.tree import DecisionTreeClassifier

# Initialize the Decision Tree Classifier
model3 = DecisionTreeClassifier(random_state=42)

# Fit the model
model3.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model3.predict(X_test)

# Model Accuracy
accuracy = accuracy_score(y_test, y_pred)

# Evaluate the model
print("Accuracy: ", accuracy * 100)
print("\nClassification Report: \n", classification_report(y_test, y_pred))
```



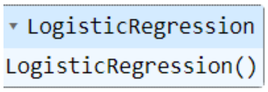
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
```



```
model1=LogisticRegression()
```

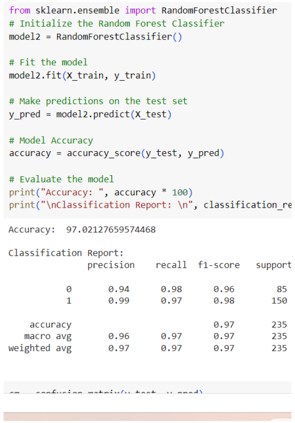
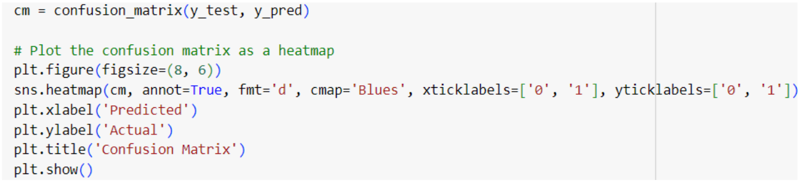
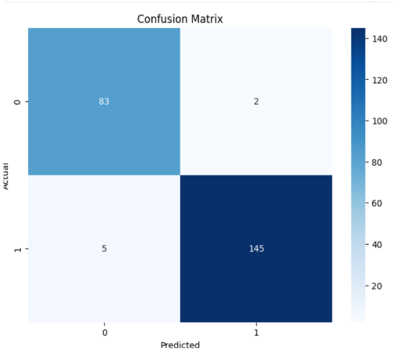
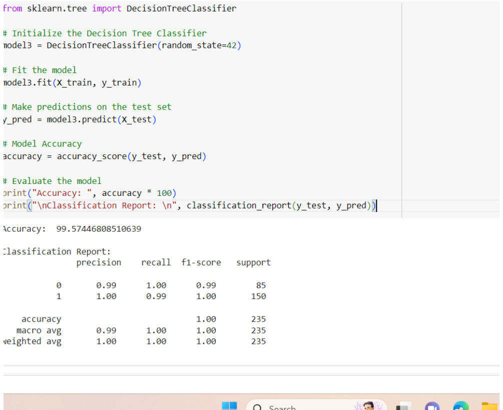
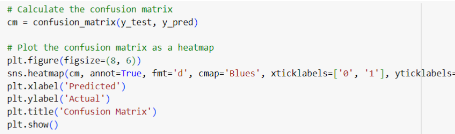


```
model1.fit(X_train,y_train)
```



```
▾ LogisticRegression
LogisticRegression()
```

Model Validation and Evaluation Report:

Model	Classification Report	F1 Scor e	Confusion Matrix
Random Forest		97%	 
Decision Tree		99%	

```
from sklearn.ensemble import RandomForestClassifier
# Initialize the Random Forest Classifier
model2 = RandomForestClassifier()

# Fit the model
model2.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model2.predict(X_test)

# Model Accuracy
accuracy = accuracy_score(y_test, y_pred)

# Evaluate the model
print("Accuracy: ", accuracy * 100)
print("\nClassification Report: \n", classification_re
```

```
Accuracy:  97.02127659574468

Classification Report:
               precision    recall  f1-score   support

           0       0.94      0.98      0.96        85
           1       0.99      0.97      0.98       150

    accuracy                           0.97       235
   macro avg       0.96      0.97      0.97       235
weighted avg       0.97      0.97      0.97       235
```

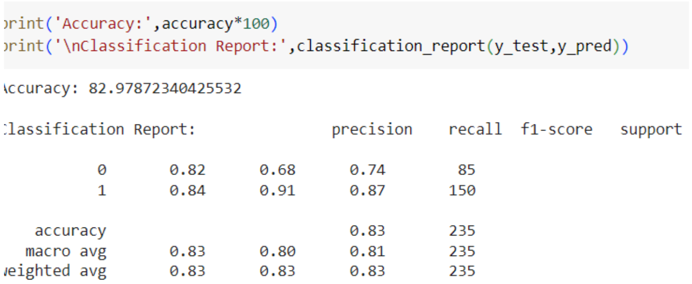
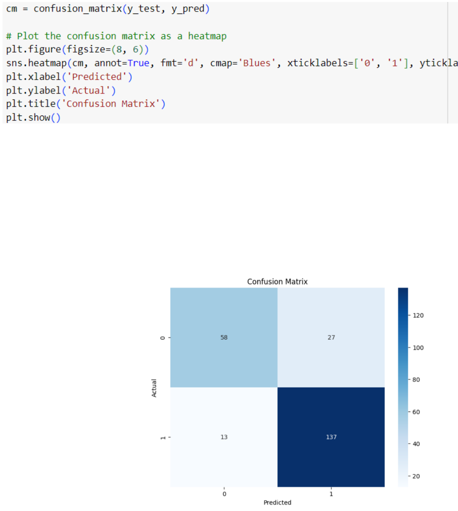
```
cm = confusion_matrix(y_test, y_pred)
```

```
cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['0', '1'], yticklabels=['0', '1'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

```
from sklearn.tree import DecisionTreeClassifier

# Initialize the Decision Tree Classifier
model3 = DecisionTreeClassifier(random_state=42)

# Fit the model
model3.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model3.predict(X_test)

# Model Accuracy
accuracy = accuracy_score(y_test, y_pred)

# Evaluate the model
print("Accuracy: ", accuracy * 100)
print("\nClassification Report: \n", classification_report(y_test, y_pred))
```

```
Accuracy:  99.57446808510639

Classification Report:
               precision    recall  f1-score   support

           0       0.99      1.00      0.99        85
           1       1.00      0.99      1.00       150

    accuracy                           1.00       235
   macro avg       0.99      1.00      1.00       235
weighted avg       1.00      1.00      1.00       235
```

```
# Calculate the confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['0', '1'], yticklabels=
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

Logistic Regression		82%	

```
print('Accuracy:',accuracy*100)
print('\nClassification Report:',classification_report(y_test,y_pred))
```

```
Accuracy: 82.97872340425532

Classification Report:               precision    recall  f1-score   support

           0       0.82      0.68      0.74        85
           1       0.84      0.91      0.87       150

    accuracy                           0.83       235
   macro avg       0.83      0.80      0.81       235
weighted avg       0.83      0.83      0.83       235
```

```
cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['0', '1'], yticklа
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```