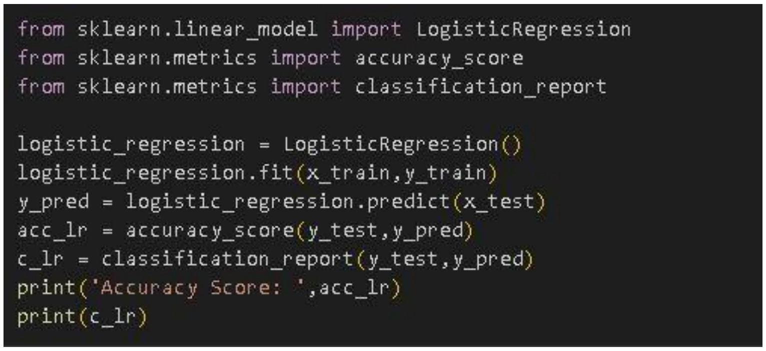
Model Development Phase

Date	1 August 2025
Skillwallet ID	SWUID20250194750
Project Title	Anemia Sense: Leveraging Machine Learning For Precise Anemia
Maximum Marks	4 Marks

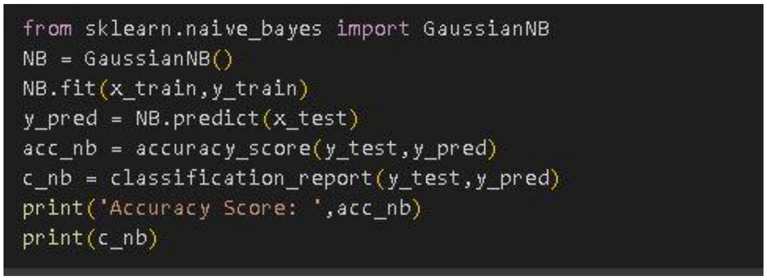
Initial Model Training Code, Model Validation and Evaluation Report

Theinitialmodeltrainingcodewillbeshowcasedinthefuturethroughascreenshot.Themodel validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

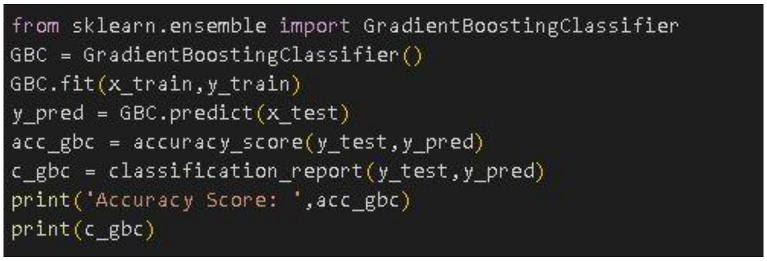
Initial Model Training Code:



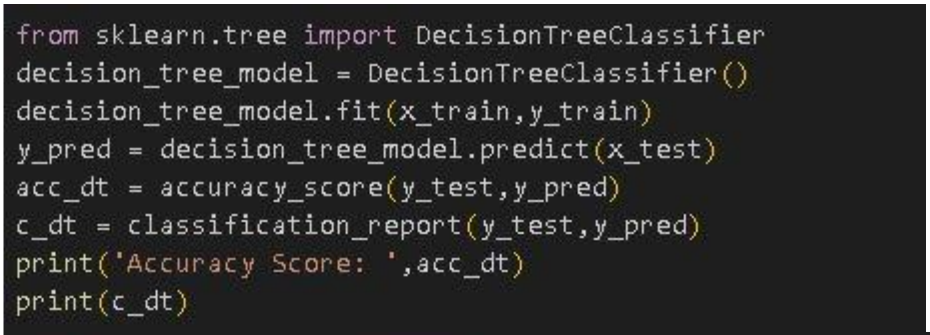
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report

logistic_regression = LogisticRegression()
logistic_regression.fit(x_train,y_train)
y_pred = logistic_regression.predict(x_test)
acc_lr = accuracy_score(y_test,y_pred)
c_lr = classification_report(y_test,y_pred)
print('Accuracy Score: ',acc_lr)
print(c_lr)
```



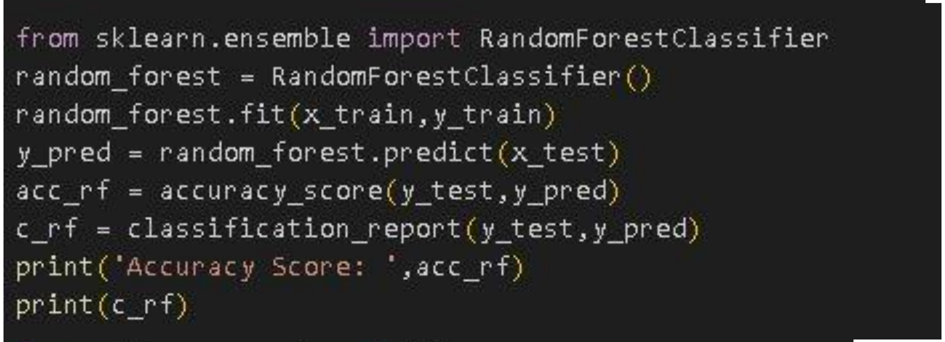
```
from sklearn.naive_bayes import GaussianNB
NB = GaussianNB()
NB.fit(x_train,y_train)
y_pred = NB.predict(x_test)
acc_nb = accuracy_score(y_test,y_pred)
c_nb = classification_report(y_test,y_pred)
print('Accuracy Score: ',acc_nb)
print(c_nb)
```



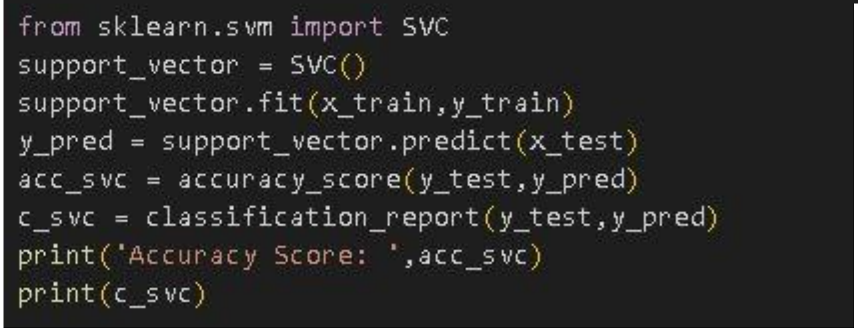
```
from sklearn.ensemble import GradientBoostingClassifier
GBC = GradientBoostingClassifier()
GBC.fit(x_train,y_train)
y_pred = GBC.predict(x_test)
acc_gbc = accuracy_score(y_test,y_pred)
c_gbc = classification_report(y_test,y_pred)
print('Accuracy Score: ',acc_gbc)
print(c_gbc)
```



```
from sklearn.tree import DecisionTreeClassifier
decision_tree_model = DecisionTreeClassifier()
decision_tree_model.fit(x_train,y_train)
y_pred = decision_tree_model.predict(x_test)
acc_dt = accuracy_score(y_test,y_pred)
c_dt = classification_report(y_test,y_pred)
print('Accuracy Score: ',acc_dt)
print(c_dt)
```

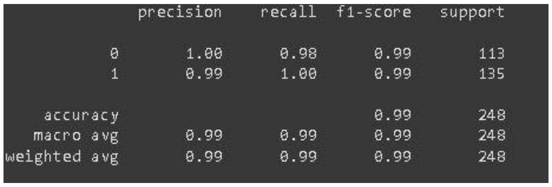
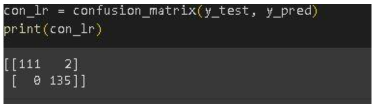


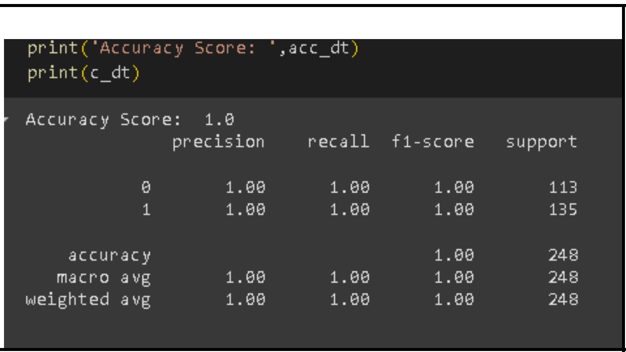
```
from sklearn.ensemble import RandomForestClassifier
random_forest = RandomForestClassifier()
random_forest.fit(x_train,y_train)
y_pred = random_forest.predict(x_test)
acc_rf = accuracy_score(y_test,y_pred)
c_rf = classification_report(y_test,y_pred)
print('Accuracy Score: ',acc_rf)
print(c_rf)
```

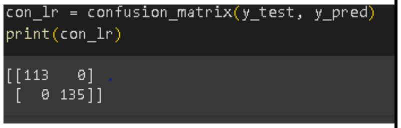


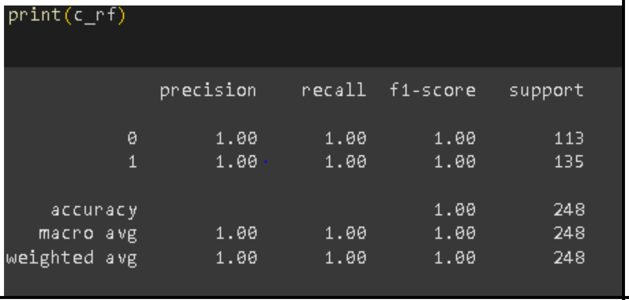
```
from sklearn.svm import SVC
support_vector = SVC()
support_vector.fit(x_train,y_train)
y_pred = support_vector.predict(x_test)
acc_svc = accuracy_score(y_test,y_pred)
c_svc = classification_report(y_test,y_pred)
print('Accuracy Score: ',acc_svc)
print(c_svc)
```

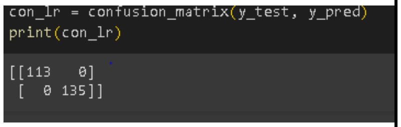
Model Validation and Evaluation Report:

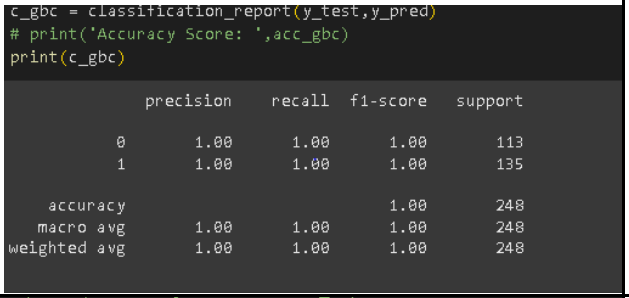
Model	Classification Report	F1 Scor e	Confusion Matrix
Linear Regression	 precision recall f1-score support 0 1.00 0.98 0.99 113 1 0.99 1.00 0.99 135 accuracy 0.99 248 macro avg 0.99 0.99 0.99 248 weighted avg 0.99 0.99 0.99 248	99%	 con_lr = confusion_matrix(y_test, y_pred) print(con_lr) [[111 2] [0 135]]

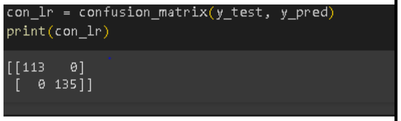
<table>
<tr>
<td>Decision Tree</td>
<td>


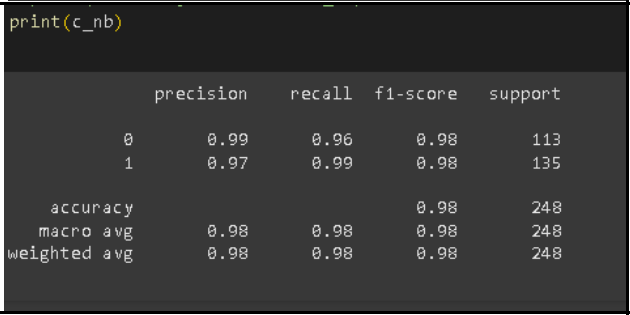
```
print('Accuracy Score: ',acc_dt)
print(c_dt)

Accuracy Score:  1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       113
           1       1.00      1.00      1.00       135

    accuracy                           1.00       248
   macro avg       1.00      1.00      1.00       248
weighted avg       1.00      1.00      1.00       248
```
</td>
<td>100 %</td>
<td>


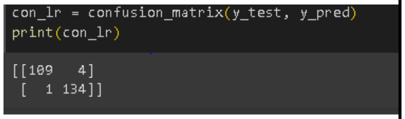
```
con_lr = confusion_matrix(y_test, y_pred)
print(con_lr)

[[113   0]
 [  0 135]]
```
</td>
</tr>
<tr>
<td>Random Forest</td>
<td>


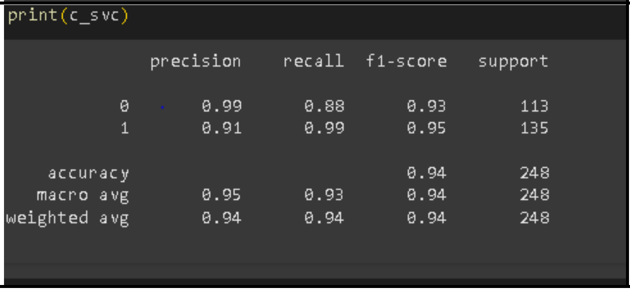
```
print(c_rf)

              precision    recall  f1-score   support

           0       1.00      1.00      1.00       113
           1       1.00      1.00      1.00       135

    accuracy                           1.00       248
   macro avg       1.00      1.00      1.00       248
weighted avg       1.00      1.00      1.00       248
```
</td>
<td>100 %</td>
<td>


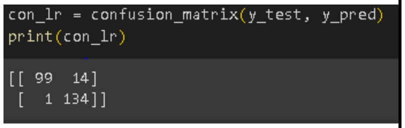
```
con_lr = confusion_matrix(y_test, y_pred)
print(con_lr)

[[113   0]
 [  0 135]]
```
</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>


```
c_gbc = classification_report(y_test,y_pred)
# print('Accuracy Score: ',acc_gbc)
print(c_gbc)

              precision    recall  f1-score   support

           0       1.00      1.00      1.00       113
           1       1.00      1.00      1.00       135

    accuracy                           1.00       248
   macro avg       1.00      1.00      1.00       248
weighted avg       1.00      1.00      1.00       248
```
</td>
<td>100 %</td>
<td>


```
con_lr = confusion_matrix(y_test, y_pred)
print(con_lr)

[[113   0]
 [  0 135]]
```
</td>
</tr>
<tr>
<td>Gaussian Naive Bayes</td>
<td>


```
print(c_nb)

              precision    recall  f1-score   support

           0       0.99      0.96      0.98       113
           1       0.97      0.99      0.98       135

    accuracy                           0.98       248
   macro avg       0.98      0.98      0.98       248
weighted avg       0.98      0.98      0.98       248
```
</td>
<td>98%</td>
<td>


```
con_lr = confusion_matrix(y_test, y_pred)
print(con_lr)

[[109   4]
 [  1 134]]
```
</td>
</tr>
<tr>
<td>Support Vector Classifier</td>
<td>


```
print(c_svc)

              precision    recall  f1-score   support

           0       0.99      0.88      0.93       113
           1       0.91      0.99      0.95       135

    accuracy                           0.94       248
   macro avg       0.95      0.93      0.94       248
weighted avg       0.94      0.94      0.94       248
```
</td>
<td>94%</td>
<td>


```
con_lr = confusion_matrix(y_test, y_pred)
print(con_lr)

[[ 99  14]
 [  1 134]]
```
</td>
</tr>
</table>