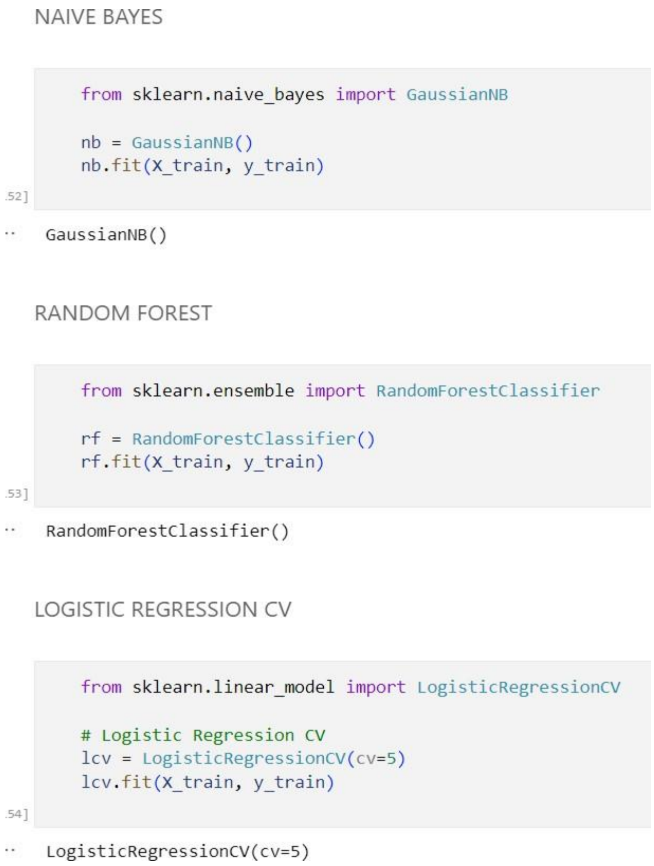
Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:



NAIVE BAYES

```
from sklearn.naive_bayes import GaussianNB

nb = GaussianNB()
nb.fit(X_train, y_train)
```

```
52]
·· GaussianNB()
```

RANDOM FOREST

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
rf.fit(X_train, y_train)
```

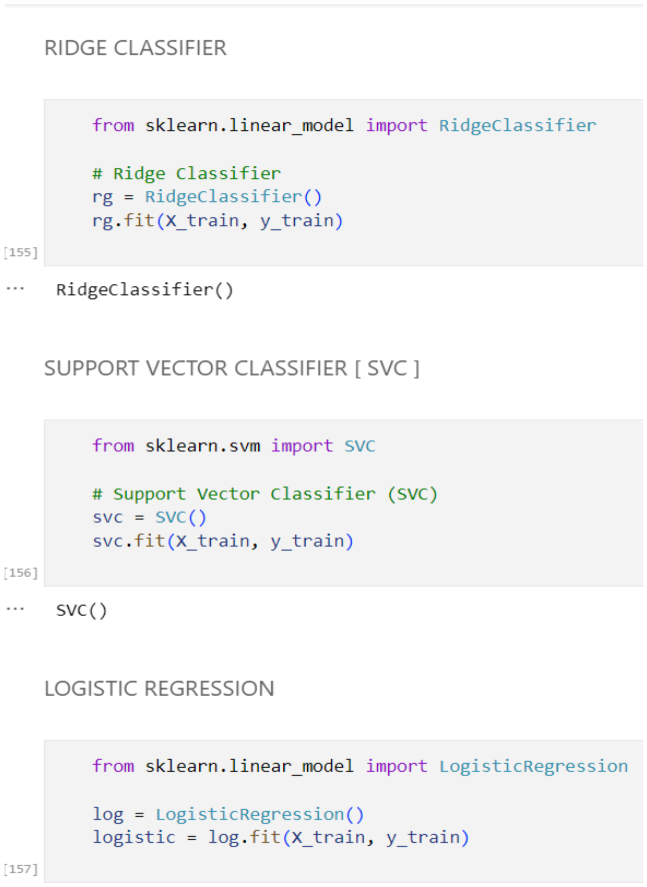
```
53]
·· RandomForestClassifier()
```

LOGISTIC REGRESSION CV

```
from sklearn.linear_model import LogisticRegressionCV

# Logistic Regression CV
lcv = LogisticRegressionCV(cv=5)
lcv.fit(X_train, y_train)
```

```
54]
·· LogisticRegressionCV(cv=5)
```



RIDGE CLASSIFIER

```
from sklearn.linear_model import RidgeClassifier

# Ridge Classifier
rg = RidgeClassifier()
rg.fit(X_train, y_train)
```

```
[155]
··· RidgeClassifier()
```

SUPPORT VECTOR CLASSIFIER [SVC]

```
from sklearn.svm import SVC

# Support Vector Classifier (SVC)
svc = SVC()
svc.fit(X_train, y_train)
```

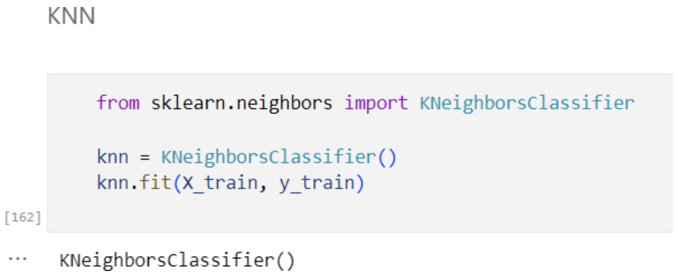
```
[156]
··· SVC()
```

LOGISTIC REGRESSION

```
from sklearn.linear_model import LogisticRegression

log = LogisticRegression()
logistic = log.fit(X_train, y_train)
```

```
[157]
```



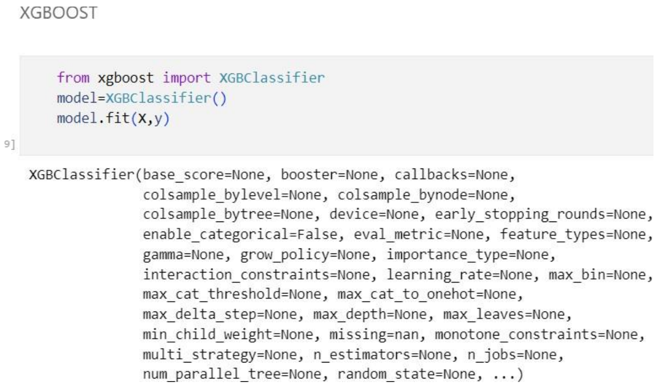
KNN

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
```

[162]

```
KNeighborsClassifier()
```



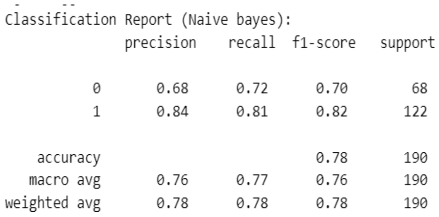
XGBOOST

```
from xgboost import XGBClassifier
model=XGBClassifier()
model.fit(X,y)
```

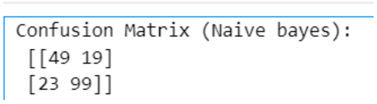
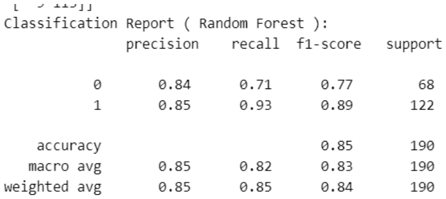
9]

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=None, n_jobs=None,
              num_parallel_tree=None, random_state=None, ...)
```

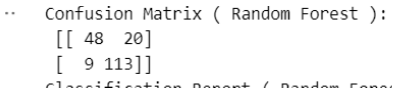
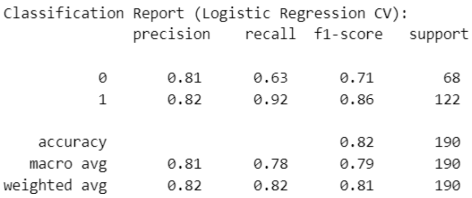
Model Validation and Evaluation Report:

<table>
<tr><th>Model</th><th>Classification Report</th><th>Accuracy</th><th>Confusion Matrix</th></tr>
<tr><td>Naive Bayes</td><td>
<pre>
Classification Report (Naive bayes):
              precision   recall f1-score   support

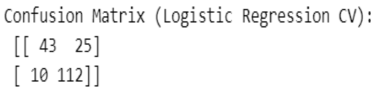
           0   0.68   0.72   0.70   68
           1   0.84   0.81   0.82   122

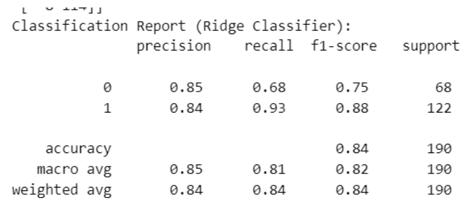
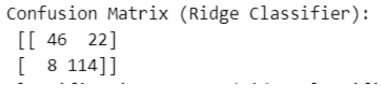
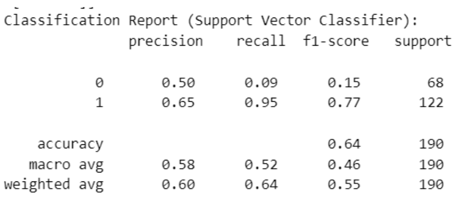
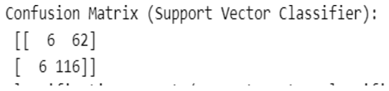
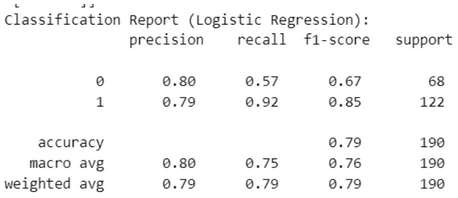
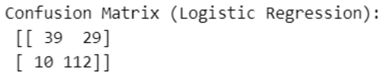
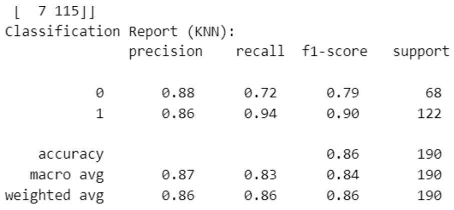
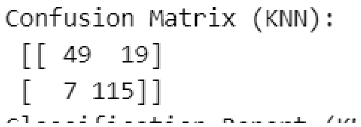
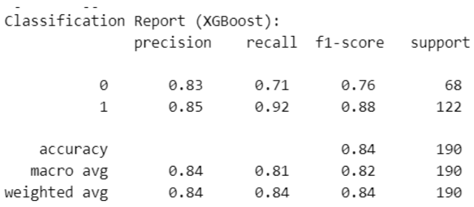
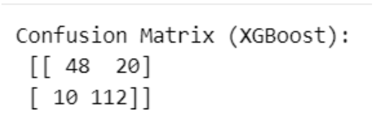
    accuracy   0.78   190
   macro avg   0.76   0.77   0.76   190
weighted avg   0.78   0.78   0.78   190
</pre></td><td>35.79 %</td><td>
<pre>
Confusion Matrix (Naive bayes):
 [[49 19]
 [23 99]]
</pre></td></tr>
<tr><td>Random Forest</td><td>
<pre>
Classification Report (Random Forest):
              precision   recall f1-score   support

           0   0.84   0.71   0.77   68
           1   0.85   0.93   0.89   122

    accuracy   0.85   190
   macro avg   0.85   0.82   0.83   190
weighted avg   0.85   0.85   0.84   190
</pre></td><td>73.16 %</td><td>
<pre>
Confusion Matrix (Random Forest):
 [[48 20]
 [9 113]]
</pre></td></tr>
<tr><td>Logistic Regression CV</td><td>
<pre>
Classification Report (Logistic Regression CV):
              precision   recall f1-score   support

           0   0.81   0.63   0.71   68
           1   0.82   0.92   0.86   122

    accuracy   0.82   190
   macro avg   0.81   0.78   0.79   190
weighted avg   0.82   0.82   0.81   190
</pre></td><td>73.16 %</td><td>
<pre>
Confusion Matrix (Logistic Regression CV):
 [[43 25]
 [10 112]]
</pre></td></tr>
</table>

Ridge Classifier	 Classification Report (Ridge Classifier): precision recall f1-score support 0 0.85 0.68 0.75 68 1 0.84 0.93 0.88 122 accuracy 0.84 190 macro avg 0.85 0.81 0.82 190 weighted avg 0.84 0.84 0.84 190	35.79 %	 Confusion Matrix (Ridge Classifier): [[46 22] [8 114]]
Support Vector Classifier	 Classification Report (Support Vector Classifier): precision recall f1-score support 0 0.50 0.09 0.15 68 1 0.65 0.95 0.77 122 accuracy 0.64 190 macro avg 0.58 0.52 0.46 190 weighted avg 0.60 0.64 0.55 190	35.79 %	 Confusion Matrix (Support Vector Classifier): [[6 62] [6 116]]
Logistic Regression	 Classification Report (Logistic Regression): precision recall f1-score support 0 0.80 0.57 0.67 68 1 0.79 0.92 0.85 122 accuracy 0.79 190 macro avg 0.80 0.75 0.76 190 weighted avg 0.79 0.79 0.79 190	74.21 %	 Confusion Matrix (Logistic Regression): [[39 29] [10 112]]
KNN	 [7 115]] Classification Report (KNN): precision recall f1-score support 0 0.88 0.72 0.79 68 1 0.86 0.94 0.90 122 accuracy 0.86 190 macro avg 0.87 0.83 0.84 190 weighted avg 0.86 0.86 0.86 190	86.32 %	 Confusion Matrix (KNN): [[49 19] [7 115]]
XG Boost	 Classification Report (XGBoost): precision recall f1-score support 0 0.83 0.71 0.76 68 1 0.85 0.92 0.88 122 accuracy 0.84 190 macro avg 0.84 0.81 0.82 190 weighted avg 0.84 0.84 0.84 190	64.21 %	 Confusion Matrix (XGBoost): [[48 20] [10 112]]