# Model Development Phase Template

| | |
|---|---|
| Date | 10 July 2024 |
| Team ID | 740006 |
| Project Title | Predictive Modeling for H1-B visa Approval Using Machine Learning |
| Maximum Marks | 10 Marks |

## Model Validation and Evaluation Report for H1-B Visa Approval

This report presents the model validation and evaluation for predicting H1-B visa approvals. We summarize the training and validation performance metrics for multiple models and include screenshots of the initial model training code, validation, and evaluation results.

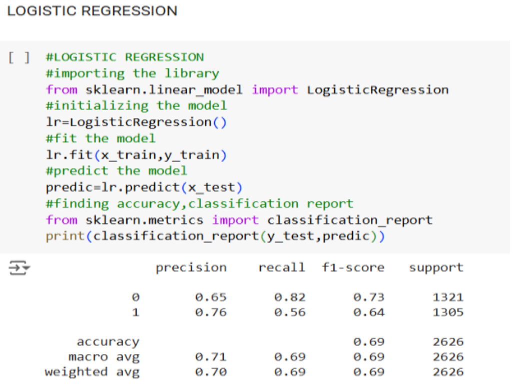
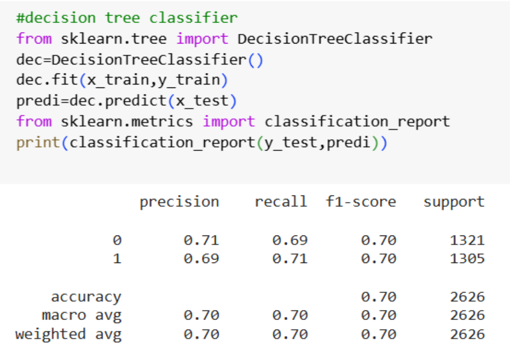
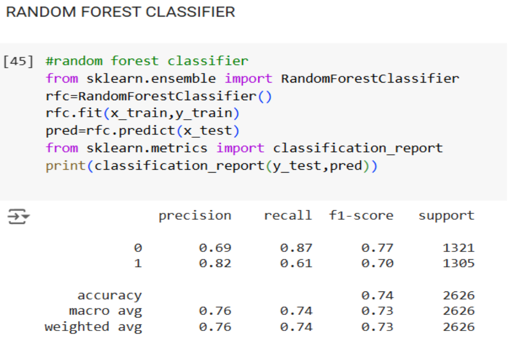
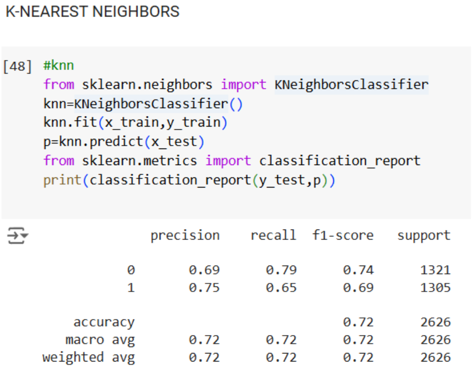
## Initial Model Training Code (5 marks):

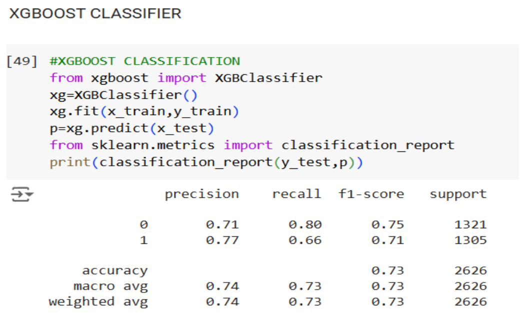
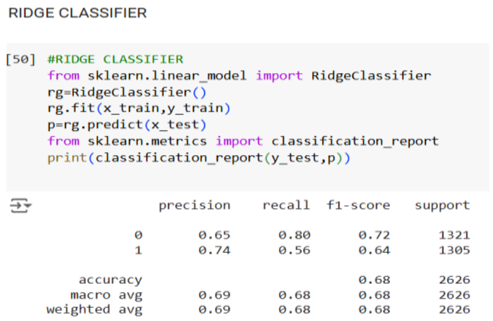
The initial model training code will be showcased through a screenshot. It includes the implementation of the Recurrent Neural Network (RNN) model with optimized hyperparameters. Key sections of the code data preprocessing, model architecture definition, training loop, and evaluation functions.

Initial Model Training Code Screenshot

## Model Validation and Evaluation Report (5 marks):

| Model | Summary | Training and Validation Performance Metrics |
|---|---|---|

| | | |
|---|---|---|
| Model 1:Logistic Regression | Logistic regression models are known for their simplicity and interpretability. They typically include accuracy, precision, recall, and F1 score to evaluate predictive performance and generalization capability. | <br>LOGISTIC REGRESSION<br>[ ] #LOGISTIC REGRESSION<br>#importing the library<br>from sklearn.linear_model import LogisticRegression<br>#initializing the model<br>lr=LogisticRegression()<br>#fit the model<br>lr.fit(x_train,y_train)<br>#predict the model<br>predic=lr.predict(x_test)<br>#finding accuracy,classification report<br>from sklearn.metrics import classification_report<br>print(classification_report(y_test,predic))<br>              precision    recall  f1-score   support<br>           0       0.65      0.82      0.73      1321<br>           1       0.76      0.56      0.64      1305<br>    accuracy                           0.69      2626<br>   macro avg       0.71      0.69      0.69      2626<br>weighted avg       0.70      0.69      0.69      2626 |
| Model 2:Decision Tree Classifier | Decision tree classifier models are intuitive and easy to interpret. They commonly include accuracy, precision, recall,  and F1 score to assess prediction accuracy and generalizability. | <br>#decision tree classifier<br>from sklearn.tree import DecisionTreeClassifier<br>dec=DecisionTreeClassifier()<br>dec.fit(x_train,y_train)<br>predi=dec.predict(x_test)<br>from sklearn.metrics import classification_report<br>print(classification_report(y_test,predi))<br>              precision    recall  f1-score   support<br>           0       0.71      0.69      0.70      1321<br>           1       0.69      0.71      0.70      1305<br>    accuracy                           0.70      2626<br>   macro avg       0.70      0.70      0.70      2626<br>weighted avg       0.70      0.70      0.70      2626 |
| Model 3:Random Forest Classifier | Random forest classifier models are ensemble mathods known for robustness. They often encompass accuracy, precision, recall, and F1 score to measure  prediction quality and robustness. | <br>RANDOM FOREST CLASSIFIER<br>[45] #random forest classifier<br>from sklearn.ensemble import RandomForestClassifier<br>rfc=RandomForestClassifier()<br>rfc.fit(x_train,y_train)<br>pred=rfc.predict(x_test)<br>from sklearn.metrics import classification_report<br>print(classification_report(y_test,pred))<br>              precision    recall  f1-score   support<br>           0       0.69      0.87      0.77      1321<br>           1       0.82      0.61      0.70      1305<br>    accuracy                           0.74      2626<br>   macro avg       0.76      0.74      0.73      2626<br>weighted avg       0.76      0.74      0.73      2626 |
| Model 4:K-Nearest Neighbors (KNN) Classifier | K-nearest neighbors classifier models rely on proximity to neighbors for classification. They typically include accuracy, precision, recall, and F1 score to evaluate prediction performance and generalization ability. | <br>K-NEAREST NEIGHBORS<br>[48] #knn<br>from sklearn.neighbors import KNeighborsClassifier<br>knn=KNeighborsClassifier()<br>knn.fit(x_train,y_train)<br>p=knn.predict(x_test)<br>from sklearn.metrics import classification_report<br>print(classification_report(y_test,p))<br>              precision    recall  f1-score   support<br>           0       0.69      0.79      0.74      1321<br>           1       0.75      0.65      0.69      1305<br>    accuracy                           0.72      2626<br>   macro avg       0.72      0.72      0.72      2626<br>weighted avg       0.72      0.72      0.72      2626 |

| | | |
|---|---|---|
| Model 5:XGBoost Classifier | XGBoost classifier models are gradient boosting algorithms known for their high performance. They typically include accuracy, precision, recall, and F1 score used to evaluate predictive performance and ability to generalize. |  |
| Model 6:Ridge Classifier | Ridge classifier models apply regularization to linear regression. Then typically include accuracy, precision, recall, F1 score, and mean squared error to evaluate prediction performance and generalization. |  |

XGBOOST CLASSIFIER

```
[49] #XGBOOST CLASSIFICATION
     from xgboost import XGBClassifier
     xg=XGBClassifier()
     xg.fit(x_train,y_train)
     p=xg.predict(x_test)
     from sklearn.metrics import classification_report
     print(classification_report(y_test,p))
```

```
              precision    recall  f1-score   support

           0       0.71      0.80      0.75      1321
           1       0.77      0.66      0.71      1305

    accuracy                           0.73      2626
   macro avg       0.74      0.73      0.73      2626
weighted avg       0.74      0.73      0.73      2626
```

RIDGE CLASSIFIER

```
[50] #RIDGE CLASSIFIER
     from sklearn.linear_model import RidgeClassifier
     rg=RidgeClassifier()
     rg.fit(x_train,y_train)
     p=rg.predict(x_test)
     from sklearn.metrics import classification_report
     print(classification_report(y_test,p))
```

```
              precision    recall  f1-score   support

           0       0.65      0.80      0.72      1321
           1       0.74      0.56      0.64      1305

    accuracy                           0.68      2626
   macro avg       0.69      0.68      0.68      2626
weighted avg       0.69      0.68      0.68      2626
```