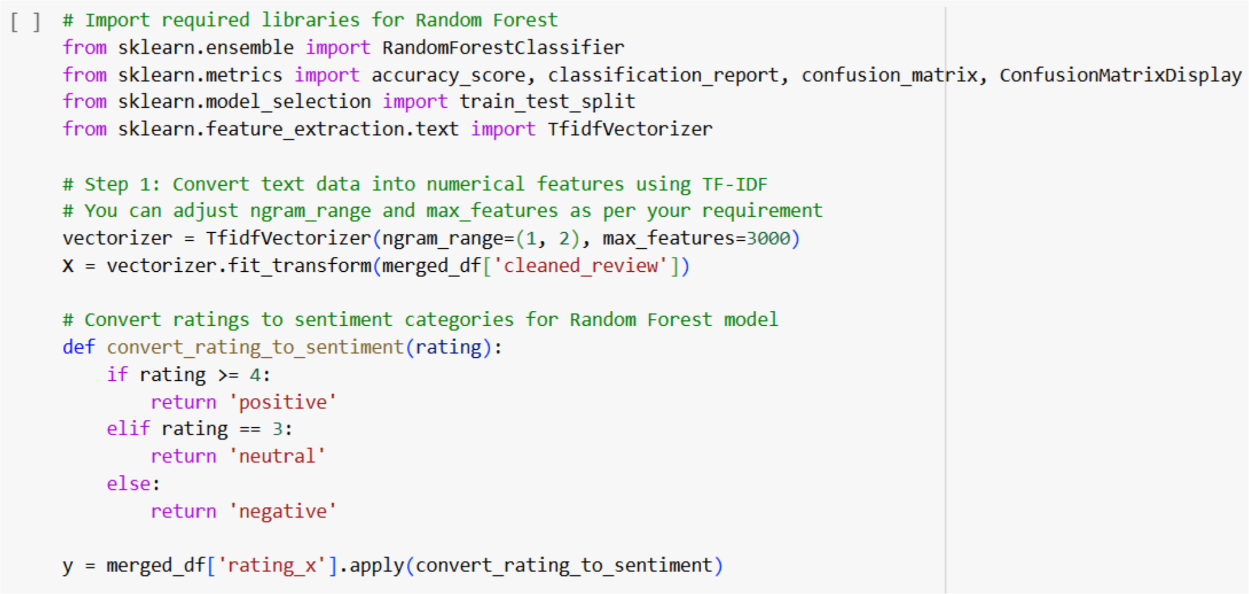
Model Development Phase Template

Date	03 October 2024
Team ID	LTVIP2024TMID24974
Project Title	Analysis Of Amazon Cell Phone Reviews
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

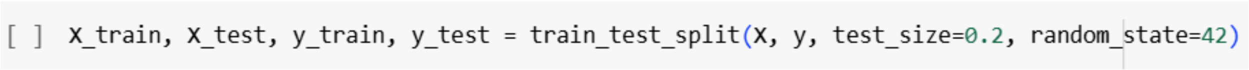
The initial model training code will be included in the final submission through screenshots. The model validation and evaluation report will summarize the performance of the Analysis of amazon cell phone reviews using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. Models like SVM, Decision Tree, Naive Bayes, Gradient Boosting and Logistic Regression will be evaluated to select the best-performing machine learning models that can be used for sentiment analysis of Amazon cell phone reviews.

Initial Model Training Code:



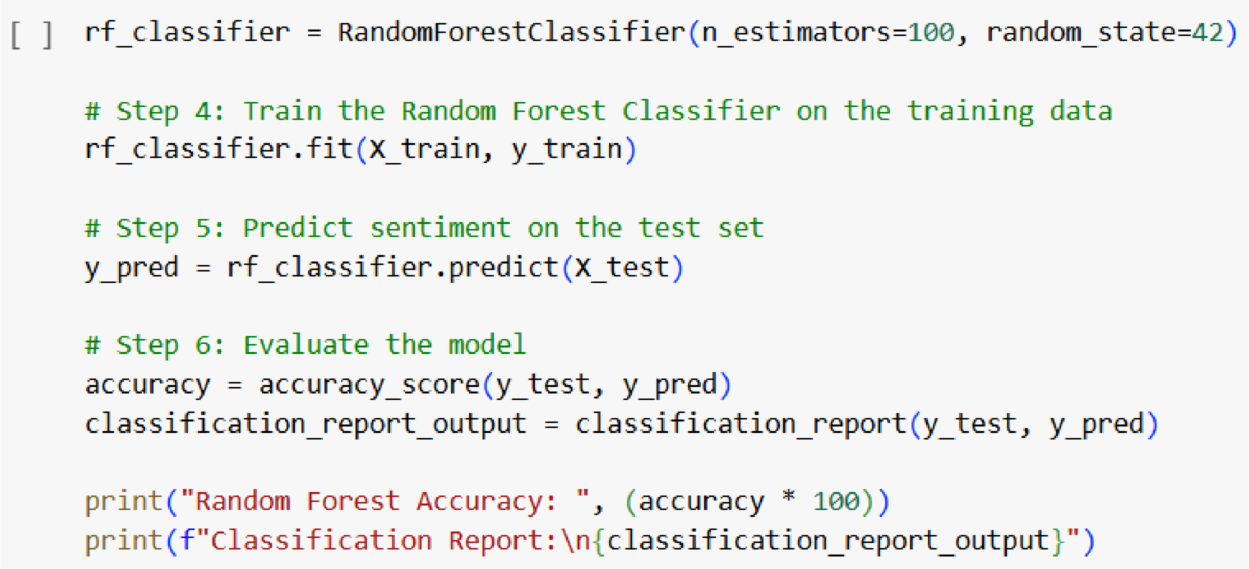
```
# Import required libraries for Random Forest
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, ConfusionMatrixDisplay
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

# Step 1: Convert text data into numerical features using TF-IDF
# You can adjust ngram_range and max_features as per your requirement
vectorizer = TfidfVectorizer(ngram_range=(1, 2), max_features=3000)
X = vectorizer.fit_transform(merged_df['cleaned_review'])

# Convert ratings to sentiment categories for Random Forest model
def convert_rating_to_sentiment(rating):
    if rating >= 4:
        return 'positive'
    elif rating == 3:
        return 'neutral'
    else:
        return 'negative'

y = merged_df['rating_x'].apply(convert_rating_to_sentiment)
```

SPLITTING THE DATA INTO TRAINING AND TESTING SETS



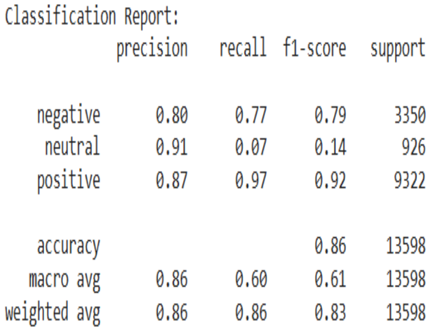
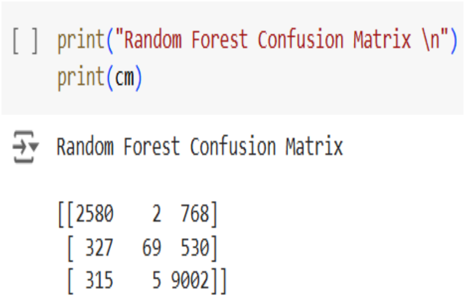
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

MODEL BUILDING (RANDOM FOREST)



```
[ ]  rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

     # Step 4: Train the Random Forest Classifier on the training data
     rf_classifier.fit(X_train, y_train)

     # Step 5: Predict sentiment on the test set
     y_pred = rf_classifier.predict(X_test)

     # Step 6: Evaluate the model
     accuracy = accuracy_score(y_test, y_pred)
     classification_report_output = classification_report(y_test, y_pred)

     print("Random Forest Accuracy: ", (accuracy * 100))
     print(f"Classification Report:\n{classification_report_output}")
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Random Forest	 Classification Report: precision recall f1-score support negative 0.80 0.77 0.79 3350 neutral 0.91 0.07 0.14 926 positive 0.87 0.97 0.92 9322 accuracy 0.86 13598 macro avg 0.86 0.60 0.61 13598 weighted avg 0.86 0.86 0.83 13598	85.68%	 [ ] print("Random Forest Confusion Matrix \n") print(cm) Random Forest Confusion Matrix [[2580 2 768] [327 69 530] [315 5 9002]]