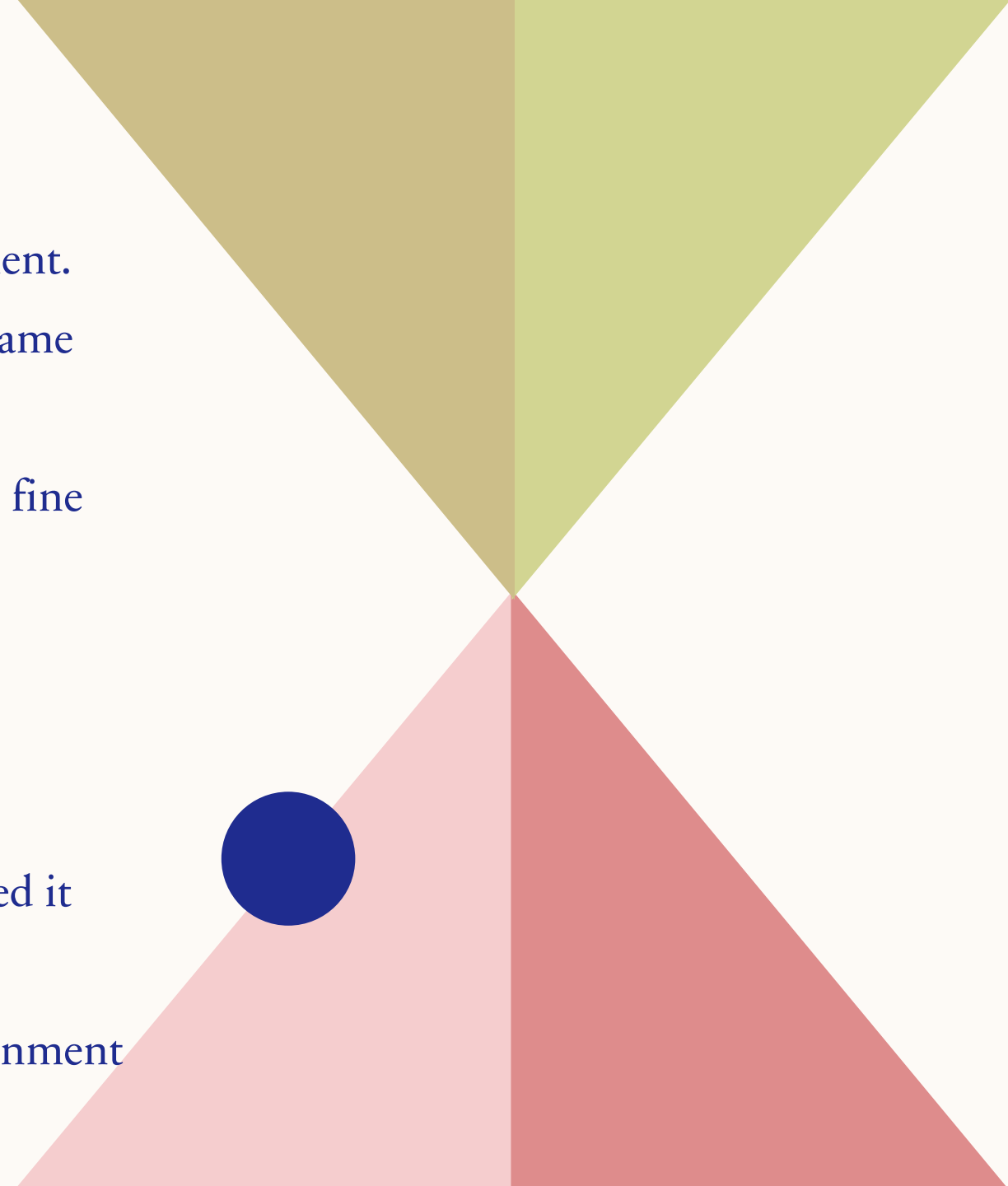
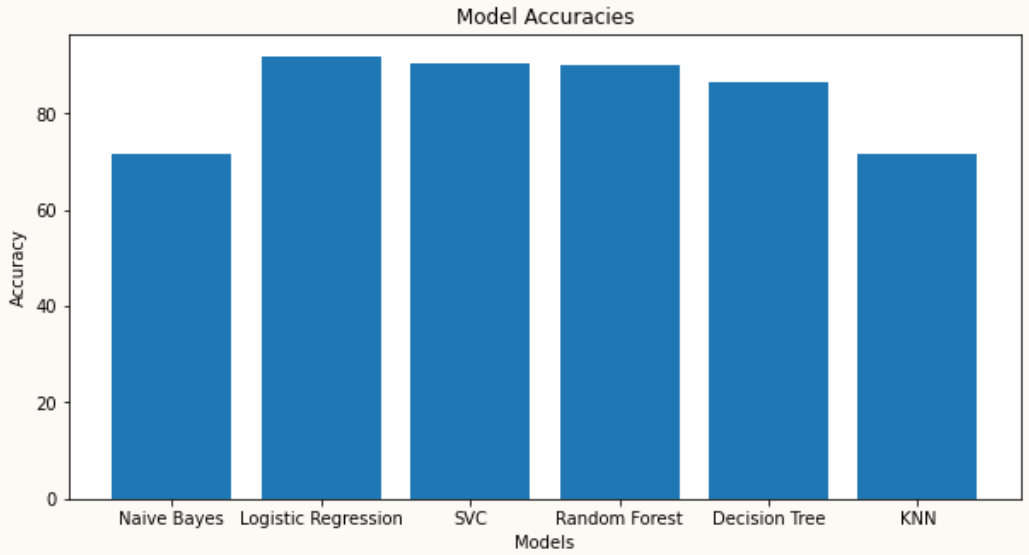
STEPS FOLLOWED

1. Built ML models for the problem statement.
2. Built Recurrent neural network for the same problem statement to prove that I have knowledge on Deep Learning too. (With fine tuning)
3. Used OOPs Concepts.
4. Developed API Server using FastAPI.
5. Integrated Swagger Documentation.
6. Containerized using Docker and deployed it on HuggingFace Spaces. (Bonus).
7. Each and every point mentioned in assignment was completed.



PERFORMANCE (BEFORE FINE TUNING)



RNN (LSTM) - ACCURACY – 90+%

	precision	recall	f1-score	support
0	0.95	0.93	0.94	1862
1	0.75	0.81	0.78	447
accuracy			0.91	2309
macro avg	0.85	0.87	0.86	2309
weighted avg	0.91	0.91	0.91	2309

BEST ML MODEL

```
Best Parameters:  {'C': 1, 'penalty': 'l2'}
Best Score:  0.9180017257348597
Best Estimator:  LogisticRegression(C=1)
Accuracy:  91.85794716327415
Confusion Matrix:  [[1801   61]
 [ 127  320]]
Classification Report:               precision    recall  f1-score   support

           0       0.93      0.97      0.95      1862
           1       0.84      0.72      0.77       447

    accuracy                           0.92      2309
   macro avg       0.89      0.84      0.86      2309
weighted avg       0.92      0.92      0.92      2309
```

Fine tuning:

All the models are duly fine tuned using GridSearchCV Algorithm and got the best hyperparameters.

Using of regularization and dropout ensured us models which are not overfitting.

```
#Logistic Regression
parameters = {'C': [0.1, 1, 10, 100, 1000], 'penalty': ['l1', 'l2']}
grid_search = GridSearchCV(LogisticRegression(), parameters, cv=5, n_jobs=-1, verbose=1)
grid_search.fit(sentiment_classifier.X_train, sentiment_classifier.y_train)
print('Best Parameters: ', grid_search.best_params_)
print('Best Score: ', grid_search.best_score_)
print('Best Estimator: ', grid_search.best_estimator_)
best_model = grid_search.best_estimator_
sentiment_classifier.train_model(best_model)
print('Accuracy: ', sentiment_classifier.get_accuracy()*100)
print('Confusion Matrix: ', sentiment_classifier.get_confusion_matrix())
print('Classification Report: ', sentiment_classifier.get_classification_report())
```

BEST MODEL AFTER TUNING

Logistic Regression – {‘C’ : 1, ‘penalty’: : ‘l2”} Score – 91.85%

Support Vector Classifier – {‘C’ : 0.1, ‘Kernel’ : ‘Linear’} Score – 91.38%

Naïve Bayes – {‘alpha’ : 0.1} Score – 90.68%

Random Forest – {‘criterion’ : ‘gini’ , ‘n_estimators’ : 300} Score – 89.95%

Decision Tree – {‘criterion’ : ‘gini’, ‘splitter’ : ‘best’} Score- 86.74%

KNN – {‘n_estimators’ : 5} Score – 71.71%

LSTM - Score – 91%