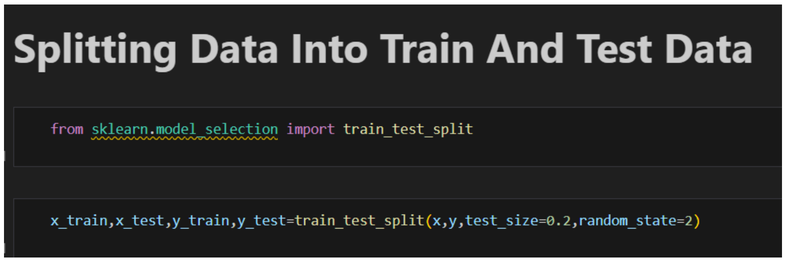
Model Development Phase Template

Date	05 June2024
Team ID	739975
Project Title	To Predict Consumer Price Index
Maximum Marks	4 Marks

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:



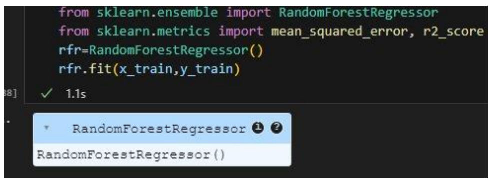
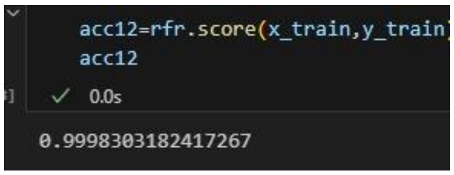
Splitting Data Into Train And Test Data

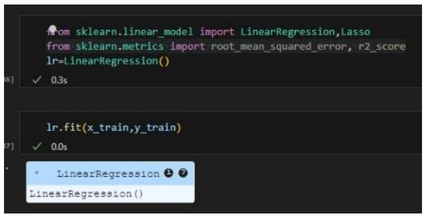
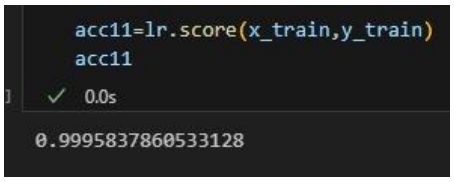
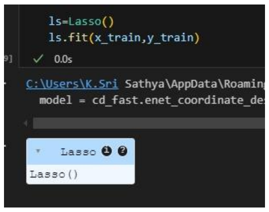
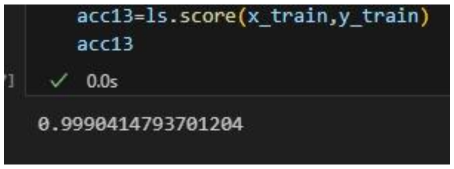
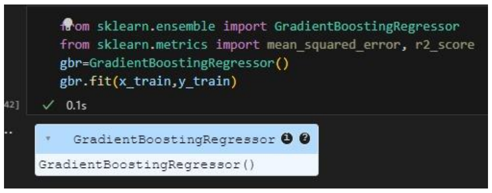
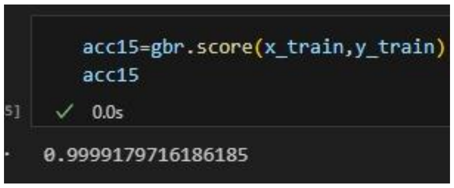
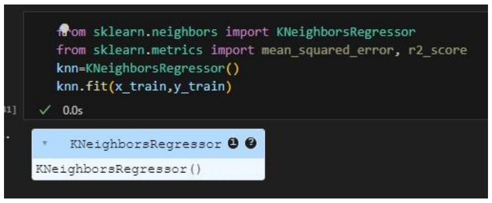
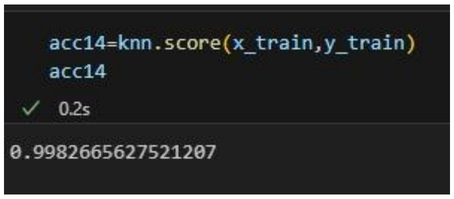
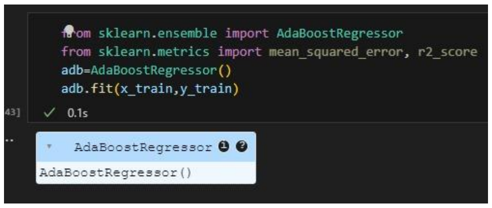
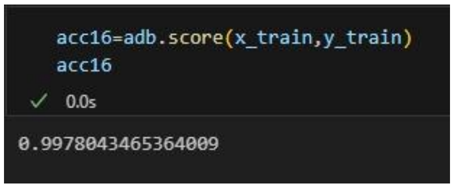
```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy

Random Forest Model	 `from sklearn.ensemble import RandomForestRegressor`
	`from sklearn.metrics import mean_squared_error, r2_score`
	`rfr=RandomForestRegressor()`
	`rfr.fit(x_train,y_train)`
	`1.1s`
	`RandomForestRegressor`
	`RandomForestRegressor()`
		 `acc12=rfr.score(x_train,y_train`
		`acc12`
		`0.0s`
		`0.9998303182417267`

Linear Regression	 `from sklearn.linear_model import LinearRegression,Lasso` `from sklearn.metrics import root_mean_squared_error, r2_score` `lr=LinearRegression()` `✓ 0.3s` `lr.fit(x_train,y_train)` `✓ 0.0s` `LinearRegression` `LinearRegression()`	 `acc11=lr.score(x_train,y_train)` `acc11` `✓ 0.0s` `0.9995837860533128`
Lasso	 `ls=Lasso()` `ls.fit(x_train,y_train)` `✓ 0.0s` `C:\Users\K.Sri Sathya\AppData\Roamin` `model = cd_fast.enet_coordinate_de` `Lasso` `Lasso()`	 `acc13=ls.score(x_train,y_train)` `acc13` `✓ 0.0s` `0.9990414793701204`
GradientBoosting Regressor	 `from sklearn.ensemble import GradientBoostingRegressor` `from sklearn.metrics import mean_squared_error, r2_score` `gbr=GradientBoostingRegressor()` `gbr.fit(x_train,y_train)` `✓ 0.1s` `GradientBoostingRegressor` `GradientBoostingRegressor()`	 `acc15=gbr.score(x_train,y_train)` `acc15` `✓ 0.0s` `0.9999179716186185`
Kneighbours Regressor	 `from sklearn.neighbors import KNeighborsRegressor` `from sklearn.metrics import mean_squared_error, r2_score` `knn=KNeighborsRegressor()` `knn.fit(x_train,y_train)` `✓ 0.0s` `KNeighborsRegressor` `KNeighborsRegressor()`	 `acc14=knn.score(x_train,y_train)` `acc14` `✓ 0.2s` `0.9982665627521207`
AdaBoost Regressor	 `from sklearn.ensemble import AdaBoostRegressor` `from sklearn.metrics import mean_squared_error, r2_score` `adb=AdaBoostRegressor()` `adb.fit(x_train,y_train)` `✓ 0.1s` `AdaBoostRegressor` `AdaBoostRegressor()`	 `acc16=adb.score(x_train,y_train)` `acc16` `✓ 0.0s` `0.9978043465364009`