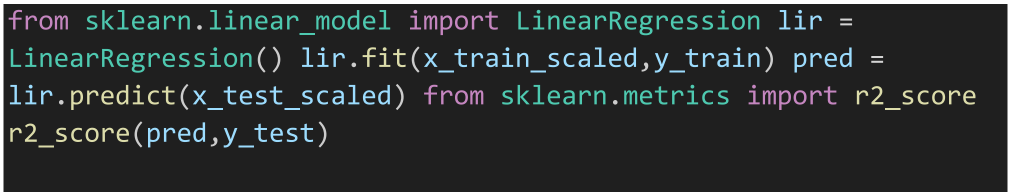
Model Development Phase Template

Date	July 2024
Team ID	739871
Project Title	Smart Home Temperature prediction using Machine Learning
Maximum Marks	10 Marks

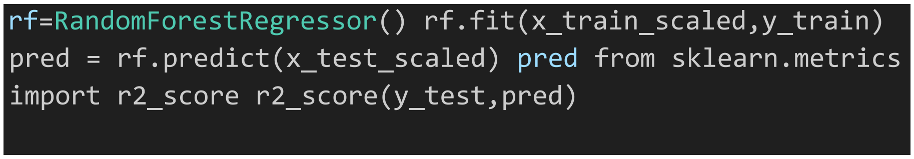
Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

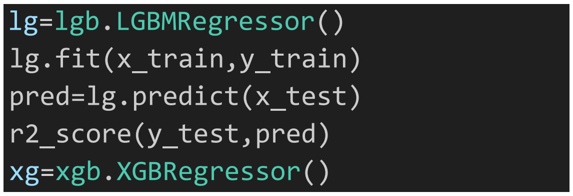
Initial Model Training Code (5 marks):



```
from sklearn.linear_model import LinearRegression lir =
LinearRegression() lir.fit(x_train_scaled,y_train) pred =
lir.predict(x_test_scaled) from sklearn.metrics import r2_score
r2_score(pred,y_test)
```



```
rf=RandomForestRegressor() rf.fit(x_train_scaled,y_train)
pred = rf.predict(x_test_scaled) pred from sklearn.metrics
import r2_score r2_score(y_test,pred)
```

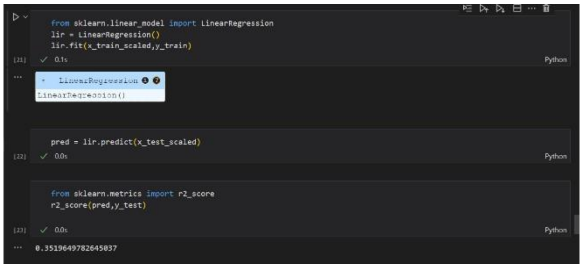


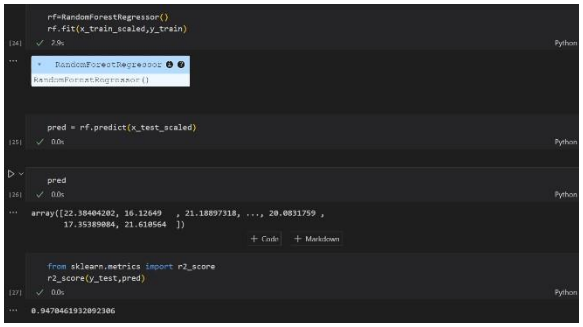
```
lg=lgb.LGBMRegressor()
lg.fit(x_train,y_train)
pred=lg.predict(x_test)
r2_score(y_test,pred)
xg=xgb.XGBRegressor()
```

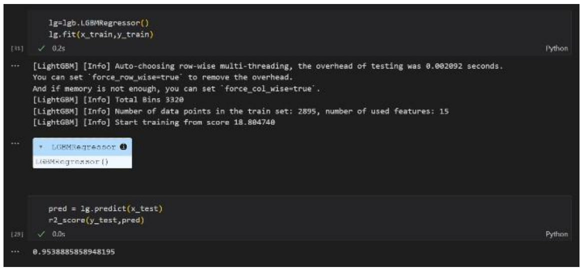
```
xg.fit(x_train,y_train)
pred=xg.predict(x_test)
r2_score(y_test,pred)
```

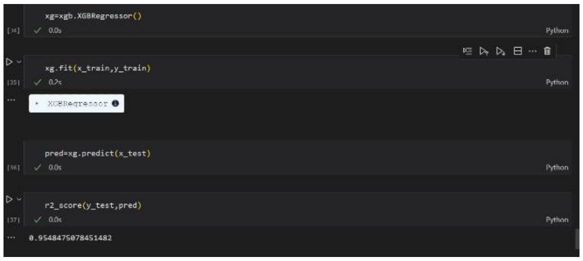
Model Evaluation and Validation Report(5 marks):

Model	Summary	Training and Validation Performance Metrics

Model 1	Linear Regressor model typically that assumes a linear relationship between input variables and temperature.	
		`from sklearn.linear_model import LinearRegression`
		`lir = LinearRegression()`
		`lir.fit(x_train_scaled,y_train)`
		`LinearRegression()`
		`pred = lir.predict(x_test_scaled)`
		`from sklearn.metrics import r2_score`
		`r2_score(pred,y_test)`
		`0.3519649782645037`

Model 2	Random forest classifier, an ensemble learning method that builds multiple decision trees and merges them together to get a more accurate prediction.	
		`rf=RandomForestRegressor()`
		`rf.fit(x_train_scaled,y_train)`
		`RandomForestRegressor()`
		`pred = rf.predict(x_test_scaled)`
		`pred`
		`array([22.38404202, 16.12649   , 21.18897318, ..., 20.0831759 ,`
		`       17.35389084, 21.610564  ])`
		`from sklearn.metrics import r2_score`
		`r2_score(y_test,pred)`
		`0.9470461932092306`

Model 3	LGBM Regressor LightGBM, a gradient boosting framework that uses tree-based learning algorithms, known for its efficiency and speed.	
		`lg=lgb.LGBMRegressor()`
		`lg.fit(x_train,y_train)`
		`[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.002092 seconds.`
		`You can set 'force_row_wise=true' to remove the overhead.`
		`And if memory is not enough, you can set 'force_col_wise=true'.`
		`[LightGBM] [Info] Total Bins 3320`
		`[LightGBM] [Info] Number of data points in the train set: 2895, number of used features: 15`
		`[LightGBM] [Info] Start training from score 18.804740`
		`LGBMRegressor()`
		`pred = lg.predict(x_test)`
		`r2_score(y_test,pred)`
		`0.9538885858948195`

Model 4	XGB Regressor, an optimized distributed gradient boosting library designed to be highly efficient and portable.	

```
xg=xgb.XGBRegressor()
[34] ✓ 0.0s                                   Python

xg.fit(x_train,y_train)
[35] ✓ 0.2s                                   Python
...  ▸ XGBRegressor

pred=xg.predict(x_test)
[36] ✓ 0.0s                                   Python

r2_score(y_test,pred)
[37] ✓ 0.0s                                   Python
...  0.9548475078451482
```