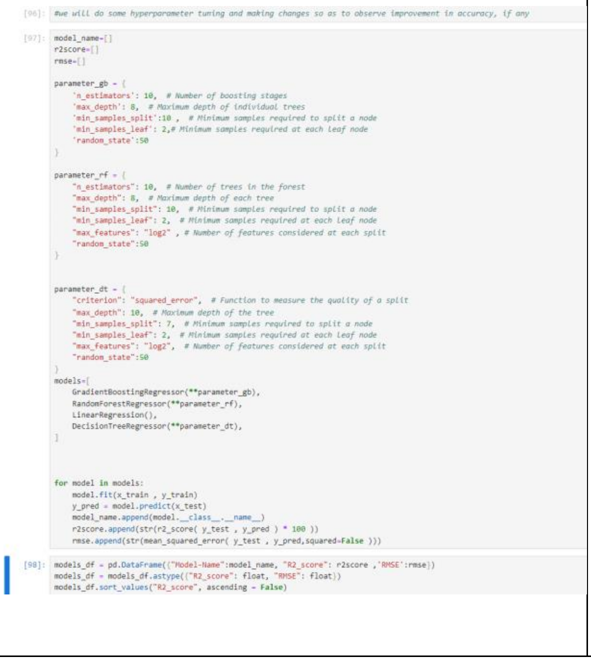
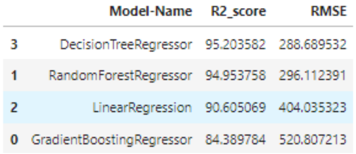
Model Optimization and Tuning Phase Template

Date	12 July 2024
Team ID	SWTID1720108739
Project Title	**Predicting The Energy Output Of Wind Turbine Based On Weather Condition**
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

5.1 Hyperparameter Tuning Documentation (6 Marks):

Model	Tuned Hyperparameters	Optimal Values
Random forest Decision Tree Gradient Boosting		

```
[96]: #we will do some hyperparameter tuning and making changes so as to observe improvement in accuracy, if any

[97]: model_name=[]
      r2score=[]
      rmse=[]

      parameter_gb = {
          'n_estimators': 10,  # Number of boosting stages
          'max_depth': 8,  # Maximum depth of individual trees
          'min_samples_split':10 ,  # Minimum samples required to split a node
          'min_samples_leaf': 2,# Minimum samples required at each leaf node
          'random_state':50
      }

      parameter_rf = {
          "n_estimators": 10,  # Number of trees in the forest
          "max_depth": 8,  # Maximum depth of each tree
          "min_samples_split": 10,  # Minimum samples required to split a node
          "min_samples_leaf": 2,  # Minimum samples required at each leaf node
          "max_features": "log2" , # Number of features considered at each split
          "random_state":50
      }


      parameter_dt = {
          "criterion": "squared_error",  # Function to measure the quality of a split
          "max_depth": 10,  # Maximum depth of the tree
          "min_samples_split": 7,  # Minimum samples required to split a node
          "min_samples_leaf": 2,  # Minimum samples required at each leaf node
          "max_features": "log2",  # Number of features considered at each split
          "random_state":50
      }
      models=[
          GradientBoostingRegressor(**parameter_gb),
          RandomForestRegressor(**parameter_rf),
          LinearRegression(),
          DecisionTreeRegressor(**parameter_dt),
      ]



      for model in models:
          model.fit(x_train , y_train)
          y_pred = model.predict(x_test)
          model_name.append(model.__class__.__name__)
          r2score.append(str(r2_score( y_test , y_pred ) * 100 ))
          rmse.append(str(mean_squared_error( y_test , y_pred,squared=False )))

[98]: models_df = pd.DataFrame({"Model-Name":model_name, "R2_score": r2score ,'RMSE':rmse})
      models_df = models_df.astype({"R2_score": float, "RMSE": float})
      models_df.sort_values("R2_score", ascending = False)
```

```
                  Model-Name   R2_score        RMSE
3      DecisionTreeRegressor  95.203582  288.689532
1      RandomForestRegressor  94.953758  296.112391
2           LinearRegression  90.605069  404.035323
0  GradientBoostingRegressor  84.389784  520.807213
```

5.2 Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric
Random forest	97.379044	94.953758
Decision Tree	95.034559	95.203582
Gradient Boosting	94.679787	84.389784
Linear regression	90.605069	90.605069

5.3 Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Random forest	Random Forests achieve higher accuracy (97% before hyperparameter tuning 94%) for wind turbine energy prediction due to ensemble averaging, robust handling of non-linear relationships, feature importance ranking, and resilience to overfitting compared to single Decision Trees, Gradient Boosting, and linear models.