In [1]:
```
import pymongo
import pandas as pd
from pymongo import MongoClient
client = pymongo.MongoClient("mongodb://localhost:27017")
```

In [2]:
```
df = pd.read_excel(r'/home/cis6180/Downloads/bottle.xlsx')
```

In [3]:
```
records = df.to_dict('records')
```

In [4]:
```
db = client['Database_Bottle']
collection = db['Collection_Bottle']
collection.insert_many(records)
```

Out[4]:
```
<pymongo.results.InsertManyResult at 0x7fbd70de8eb0>
```

In [5]:
```
entry = collection.find({}, {'_id':0, 'Salnty': 1, 'T_degC': 1})
```

In [6]:
```
dFRame = pd.DataFrame(list(entry))
```

In [7]:
```
pandas_df=pd.DataFrame()
pandas_df[['Salnty', 'T_degC']] = dFRame[['Salnty', 'T_degC']]
```

In [8]:
```
from pyspark.sql import SparkSession
spark = SparkSession.builder \
        .appName("Spark session in Regression")  \
        .getOrCreate()
```

```
23/04/06 04:06:57 WARN Utils: Your hostname, cis6180 resolves to a loo
pback address: 127.0.1.1; using 10.0.2.15 instead (on interface enp0s
3)
23/04/06 04:06:57 WARN Utils: Set SPARK_LOCAL_IP if you need to bind t
o another address

Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
setLogLevel(newLevel).

23/04/06 04:06:58 WARN NativeCodeLoader: Unable to load native-hadoop
library for your platform... using builtin-java classes where applicab
le
23/04/06 04:06:59 WARN Utils: Service 'SparkUI' could not bind on port
4040. Attempting port 4041.
```

In [ ]:

In [9]:
```
#convert pandas DataFrame to Spark DataFrame
spark_Dataf = spark.createDataFrame(pandas_df)
```

In [10]:
```
sp_df = spark_Dataf.dropna()
```

In [11]:
```
train_df,test_df = sp_df.randomSplit([0.75,0.25], seed=42)
```

In [12]:
```
from pyspark.ml import Pipeline
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.regression import RandomForestRegressor, GBTRegressor
from pyspark.ml.evaluation import RegressionEvaluator
from pyspark.ml.tuning import CrossValidator, ParamGridBuilder
from pyspark.sql.functions import rand

# Define the VectorAssembler to create the feature vector
Vec_assembler = VectorAssembler(inputCols=["Salnty"], outputCol="featur

# Define the RandomForestRegressor and GBTRegressor models
Randomf = RandomForestRegressor(featuresCol="features", labelCol="T_deg
Gboost = GBTRegressor(featuresCol="features", labelCol="T_degC")

# Define the parameter grids for cross-validation
rf_param_grid = ParamGridBuilder() \
    .addGrid(Randomf.numTrees, [5, 10, 20]) \
    .addGrid(Randomf.maxDepth, [2, 5, 10]) \
    .build()

gbt_param_grid = ParamGridBuilder() \
    .addGrid(Gboost.maxDepth, [2, 5, 10]) \
    .addGrid(Gboost.maxIter, [10, 20, 50]) \
    .build()

#Evaluation Metric
evalr = RegressionEvaluator(labelCol="T_degC", predictionCol="predictio

# Define the cross-validator for RandomForestRegressor
rf_cv = CrossValidator(estimator=Randomf, estimatorParamMaps=rf_param_g

# Define the cross-validator for GBTRegressor
gbt_cv = CrossValidator(estimator=Gboost, estimatorParamMaps=gbt_param_

# Define the pipeline for the RandomForestRegressor
rf_pipeline = Pipeline(stages=[Vec_assembler, rf_cv])

# Define the pipeline for the GBTRegressor
gbt_pipeline = Pipeline(stages=[Vec_assembler, gbt_cv])
```

In [13]:
```
# Fit the pipelines using the training data
rmodel_ = rf_pipeline.fit(train_df)


# Evaluate the models on the test data
rpredictions = rmodel_.transform(test_df)

rf_rmse = evalr.evaluate(rpredictions)
```

```
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.util.SizeEstima
tor$ (file:/home/cis6180/anaconda3/lib/python3.9/site-packages/pyspark
/jars/spark-core_2.12-3.3.1.jar) to field java.nio.charset.Charset.nam
e
WARNING: Please consider reporting this to the maintainers of org.apac
he.spark.util.SizeEstimator$
WARNING: Use --illegal-access=warn to enable warnings of further illeg
al reflective access operations
WARNING: All illegal access operations will be denied in a future rele
ase
```

In [14]:
```
gmodel = gbt_pipeline.fit(train_df)

gpredictions = gmodel.transform(test_df)

gbt_rmse = evalr.evaluate(gpredictions)
```

```
23/04/06 04:08:15 WARN InstanceBuilder$NativeBLAS: Failed to load impl
ementation from:dev.ludovic.netlib.blas.JNIBLAS
23/04/06 04:08:15 WARN InstanceBuilder$NativeBLAS: Failed to load impl
ementation from:dev.ludovic.netlib.blas.ForeignLinkerBLAS
```

In [15]:
```
# Define the evaluation metrics
evaluator_rmse = RegressionEvaluator(labelCol="T_degC", predictionCol="
evaluator_mae = RegressionEvaluator(labelCol="T_degC", predictionCol="p
evaluator_r2 = RegressionEvaluator(labelCol="T_degC", predictionCol="pr
evaluator_mse = RegressionEvaluator(labelCol="T_degC", predictionCol="p

# Calculate the evaluation metrics for both models
rf_rmse = evaluator_rmse.evaluate(rpredictions)
gbt_rmse = evaluator_rmse.evaluate(gpredictions)

rf_mae = evaluator_mae.evaluate(rpredictions)
gbt_mae = evaluator_mae.evaluate(gpredictions)

rf_r2 = evaluator_r2.evaluate(rpredictions)
gbt_r2 = evaluator_r2.evaluate(gpredictions)

rf_mse = evaluator_mse.evaluate(rpredictions)
gbt_mse = evaluator_mse.evaluate(gpredictions)

# Print the evaluation metrics for both models
print("--------------------------------------------------")
print("Random Forest Regressor Metrics:")
print("Value of RMSE :", rf_rmse)
print("Value of R-squared:", rf_r2)
print("Value of MAE:", rf_mae)
print("Value of MSE:", rf_mse)

print("--------------------------------------------------")

print("\nGradient-Boosted Tree Regressor Metrics:")
print("Value of RMSE:", gbt_rmse)
print("Value of MAE:", gbt_mae)
print("Value of R-squared:", gbt_r2)
print("VAlue of MSE:", gbt_mse)
print("--------------------------------------------------")
```

```
--------------------------------------------------
Random Forest Regressor Metrics:
Value of RMSE : 2.38589395763419
Value of R-squared: 0.6431987287515488
Value of MAE: 1.7742396324496303
Value of MSE: 5.692489977075337
--------------------------------------------------

Gradient-Boosted Tree Regressor Metrics:
Value of RMSE: 2.387401839864875
Value of MAE: 1.7742766736476665
Value of R-squared: 0.642747590252885
VAlue of MSE: 5.699687544990191
--------------------------------------------------
```

In [ ]: