

```
In [48]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [49]: df=pd.read_csv('/content/drive/My Drive/IBM Hack2020/T1.csv')
```

```
In [50]: df.drop(columns=df[['Date/Time','Theoretical_Power_Curve (KWh)']],inplace=True)
y=df['LV ActivePower (kW)']
df.drop(columns=['LV ActivePower (kW)'],axis=1,inplace=True)
```

```
In [56]: df['Wind Direction (°)']=(df['Wind Direction (°)']-df['Wind Direction (°)'].mean())/(df['Wind Direction (°)'].std())
df['Wind Speed (m/s)']=(df['Wind Speed (m/s)']-df['Wind Speed (m/s)'].mean())/(df['Wind Speed (m/s)'].std())
```

```
In [53]: y_train=y[:42283]
y_test=y[42283:]
X_train=df.iloc[:42283]
X_test=df.iloc[42283:]
```

```
In [54]: from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor
from sklearn.metrics import accuracy_score,r2_score,mean_squared_error
xgr=XGBRegressor()
rf=RandomForestRegressor()
lr=LinearRegression()
dt=DecisionTreeRegressor()
sm=SVR()
```

```

model_xg=xgr.fit(X_train,y_train)
y_xg=model_xg.predict(X_test)
model_rf=rf.fit(X_train,y_train)
y_rf=model_rf.predict(X_test)
model_lr=lr.fit(X_train,y_train)
y_lr=model_lr.predict(X_test)
model_dt=dt.fit(X_train,y_train)
y_dt=model_dt.predict(X_test)
model_sm=sm.fit(X_train,y_train)
y_sm=model_sm.predict(X_test)

print('R2-xgb',r2_score(y_test,y_xg))
print('RMSE-xgb',np.sqrt(mean_squared_error(y_test,y_xg)))

print('R2-rf',r2_score(y_test,y_rf))
print('RMSE-rf',np.sqrt(mean_squared_error(y_test,y_rf)))

print('R2-lr',r2_score(y_test,y_lr))
print('RMSE-lr',np.sqrt(mean_squared_error(y_test,y_lr)))

print('R2-dt',r2_score(y_test,y_dt))
print('RMSE-dt',np.sqrt(mean_squared_error(y_test,y_dt)))

print('R2-svm',r2_score(y_test,y_sm))
print('RMSE-svm',np.sqrt(mean_squared_error(y_test,y_sm)))

```

[17:20:19] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.

```

R2-xgb 0.8652235636163211
RMSE-xgb 504.17752712506893
R2-rf 0.8169912962483032
RMSE-rf 587.5060886041389
R2-lr 0.8184357809835172
RMSE-lr 585.1829072925026
R2-dt 0.7185884877047644
RMSE-dt 728.529981763548
R2-svm 0.8891383750923172
RMSE-svm 457.2641138184238

```

```
In [15]: params={
    "learning_rate"      : [0.05, 0.01,0.03,0.1, 0.15, 0.2] ,
    "n_estimators"       : [50, 100, 150, 200, 500, 800,1000,1500] ,
    "max_depth"          : [ 3, 4, 5, 6, 8, 10, 12, 15,20,25],
    "min_child_weight"   : [ 1, 3, 5, 7 ,10,15,20,25],
    "gamma"              : [ 0.0, 0.1, 0.2 , 0.3, 0.4 ],
    "subsample"          : [ 0.1, 0.2 , 0.3, 0.4,0.6,0.8,1 ],
    "reg_lambda"         : [ 0.0, 0.1, 0.2 , 0.3, 0.4 ,0.6,0.8,1],
    "reg_alpha"          : [ 0.0, 0.1, 0.2 , 0.3, 0.4 ],
    "colsample_bytree"   : [ 0.3, 0.4, 0.5 , 0.7,0.9 ],
    "colsample_bylevel"  : [ 0.3, 0.4, 0.5 , 0.7,0.9 ]

}
```

```
In [55]: from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
```

```
In [17]: def timer(start_time=None):
    if not start_time:
        start_time = datetime.now()
        return start_time
    elif start_time:
        thour, temp_sec = divmod((datetime.now() - start_time).total_seconds(), 3600)
        tmin, tsec = divmod(temp_sec, 60)
        print('\n Time taken: %i hours %i minutes and %s seconds.' % (thour, tmin, round(tsec, 2)))
```

```
In [ ]: random_search=RandomizedSearchCV(xgr,param_distributions=params_rf,n_iter=10,n_jobs=-1,cv=5,verbose=3)
from datetime import datetime
start_time = timer(None) # timing starts from this point for "start_time" variable
random_search.fit(X_train,y_train)
timer(start_time) # timing ends here for "start_time" variable
```

```
In [ ]: random_search.best_estimator_
```

```
In [29]: xg=XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=0.7,
                    colsample_bynode=1, colsample_bytree=0.3, gamma=0.2,
                    importance_type='gain', learning_rate=0.03, max_delta_step=0,
                    max_depth=8, min_child_weight=25, missing=None, n_estimators=800,
                    n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
                    reg_alpha=0.2, reg_lambda=0.8, scale_pos_weight=1, seed=None,
                    silent=None, subsample=0.1, verbosity=1)
x=xgr.fit(X_train,y_train)
y1=x.predict(X_test)
r2_score(y_test,y1)
```

[15:49:50] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.

Out[29]: 0.8986270012216343

```
In [40]: r=RandomForestRegressor()
params_rf={
    "n_estimators"      : [50, 100, 150, 200, 500, 800,1000,1500] ,
    "max_depth"         : [ 3, 4, 5, 6, 8, 10, 12, 15,20,25]}
```

```
In [41]: random_search=RandomizedSearchCV(rf,param_distributions=params_rf,n_iter=10,n_jobs=-1,cv=5,verbose=3)
```

```
In [42]: from datetime import datetime
# Here we go
start_time = timer(None) # timing starts from this point for "start_time" variable
random_search.fit(X_train,y_train)
timer(start_time) # timing ends here for "start_time" variable
```

Fitting 5 folds for each of 10 candidates, totalling 50 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.

/usr/local/lib/python3.6/dist-packages/joblib/externals/loky/process_executor.py:691: UserWarning: A worker stopped while some jobs were given to the executor. This can be caused by a too short worker timeout or by a memory leak.

"timeout or by a memory leak.", UserWarning

[Parallel(n_jobs=-1)]: Done 28 tasks | elapsed: 12.0min

[Parallel(n_jobs=-1)]: Done 50 out of 50 | elapsed: 16.2min finished

```
timer(start_time) # timing ends here for start_time variable
```

Fitting 5 folds for each of 10 candidates, totalling 50 fits

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.  
/usr/local/lib/python3.6/dist-packages/joblib/externals/loky/process_executor.py:691: UserWarning: A worker stopped while some jobs were given to the  
executor. This can be caused by a too short worker timeout or by a memory leak.  
"timeout or by a memory leak.", UserWarning  
[Parallel(n_jobs=-1)]: Done 28 tasks      | elapsed: 12.0min  
[Parallel(n_jobs=-1)]: Done 50 out of 50 | elapsed: 16.2min finished  
Time taken: 0 hours 16 minutes and 29.19 seconds.
```

In [43]: `random_search.best_estimator_`

Out[43]: `RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
max_depth=4, max_features='auto', max_leaf_nodes=None,
max_samples=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1,
min_samples_split=2, min_weight_fraction_leaf=0.0,
n_estimators=500, n_jobs=None, oob_score=False,
random_state=None, verbose=0, warm_start=False)`

In [33]: `sv=SVR(gamma='auto',C=100,epsilon=0.4)
x=rf.fit(X_train,y_train)
y1=x.predict(X_test)
r2_score(y_test,y1)`

Out[33]: 0.8896894942220148