**Mercedes-Benz Greener Manufacturing**

**Implementation**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

palette = sns.color\_palette()

sns.set()

df = pd.read\_csv('C:/Users/patricia/Desktop/simplilearn/train.csv')

df.info()

y = df['y'].values

plt.figure(figsize=(15,6))

plt.hist(y,bins=20)

plt.ylabel('# of times / Frequency')

plt.xlabel('time (seconds)');

plt.figure(figsize=(15,6))

plt.plot(y[4000:4200]);

cols = [c for c in df.columns if 'X' in c]

print(f'Number of Independent Variables: {len(cols)}')

print('Feature Types:')

df[cols].dtypes.value\_counts()

\n\n counts = [[],[],[]]

for c in cols:

typ = df[c].dtype

uniq = len(np.unique(df[c]))

if uniq==1: counts[0].append(c)

elif uniq==2 and typ==np.int64: counts[1].append(c)

else: counts[2].append(c)

print (f' Constant Features:\n {counts [0]} \n\n Binary Features:\n {counts[1]} Categorical Features:\n {counts[2]}')

dft = pd.read\_csv('C:/Users/patricia/Desktop/simplilearn/test.csv')

features = list(set(df.columns)-set(['ID','y']))

X\_train = df[features]

y\_train = df['y'].values

X\_test = dft[features]

id\_test = df['ID'].values

for col in features:

cardinality = len(np.unique(X\_train[col]))

# Dropping the constant columns

if cardinality==1:

X\_train.drop(col,axis=1)

X\_test.drop(col,axis=1)

# Label Encoding using ord

if cardinality>2:

mapper = lambda x: sum(ord(digit) for digit in x)

X\_train[col] = X\_train[col].apply(mapper)

X\_test[col] = X\_test[col].apply(mapper)

ord('z')+ord('a')

df[counts[2]].head()

X\_train[counts[2]].head()

!pip install xgboost

import xgboost as xgb

from sklearn.metrics import r2\_score

from sklearn.model\_selection import train\_test\_split

X\_train, X\_valid, y\_train, y\_valid = train\_test\_split(X\_train,y\_train,test\_size=0.2,random\_state=42)

d\_train = xgb.DMatrix(X\_train,label=y\_train)

d\_valid = xgb.DMatrix(X\_valid,label=y\_valid)

d\_test = xgb.DMatrix(X\_test)

params = {}

params['Objective'] = 'reg:linear'

params['eta'] = 0.02

params['max\_depth'] = 4

def xgb\_r2score(preds,dtrain):

labels = dtrain.get\_label()

return 'r2',r2\_score(labels,preds)

watchlist = [(d\_train,'Train'),(d\_valid,'Validation')]

clf = xgb.train(params,d\_train,1000,watchlist,early\_stopping\_rounds=50,

feval=xgb\_r2score,maximize=True,verbose\_eval=10)

p\_test = clf.predict(d\_test)

pred = pd.DataFrame()

pred['ID'] = id\_test

pred['y'] = p\_test

pred.head(10)

X\_train.shape

from sklearn.decomposition import PCA

pca2 = PCA(n\_components=3)

pca2\_results = pca2.fit\_transform(X\_train)

pca2\_results.shape