In [ ]:
```
!pip install catboost
```

```
Requirement already satisfied: catboost in /usr/local/lib/python3.1
1/dist-packages (1.2.8)
Requirement already satisfied: graphviz in /usr/local/lib/python3.1
1/dist-packages (from catboost) (0.20.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.
11/dist-packages (from catboost) (3.10.0)
Requirement already satisfied: numpy<3.0,>=1.16.0 in /usr/local/lib/
python3.11/dist-packages (from catboost) (2.0.2)
Requirement already satisfied: pandas>=0.24 in /usr/local/lib/python
3.11/dist-packages (from catboost) (2.2.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/di
st-packages (from catboost) (1.14.1)
Requirement already satisfied: plotly in /usr/local/lib/python3.11/d
ist-packages (from catboost) (5.24.1)
Requirement already satisfied: six in /usr/local/lib/python3.11/dist
-packages (from catboost) (1.17.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/
lib/python3.11/dist-packages (from pandas>=0.24->catboost) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python
3.11/dist-packages (from pandas>=0.24->catboost) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/pyth
on3.11/dist-packages (from pandas>=0.24->catboost) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/py
thon3.11/dist-packages (from matplotlib->catboost) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python
3.11/dist-packages (from matplotlib->catboost) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/p
ython3.11/dist-packages (from matplotlib->catboost) (4.57.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/p
ython3.11/dist-packages (from matplotlib->catboost) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/pyt
hon3.11/dist-packages (from matplotlib->catboost) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.1
1/dist-packages (from matplotlib->catboost) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/py
thon3.11/dist-packages (from matplotlib->catboost) (3.2.3)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/pyt
hon3.11/dist-packages (from plotly->catboost) (9.1.2)
```

In [ ]:
```
import numpy as np
import pandas as pd
from functools import reduce
import matplotlib.pyplot as plt
import seaborn as sns
import time
from itertools import combinations

from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split, KFold
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostin
from sklearn.neighbors import KNeighborsRegressor
```

```
from sklearn.metrics import mean_squared_error, r2_score, mean_abso

import xgboost as xgb
from xgboost import XGBRegressor
from catboost import CatBoostRegressor
import lightgbm as lgb
from lightgbm import LGBMRegressor
import torch
```

In [ ]:
```
# Load the file with VISDATE as well
csv_file_path3 = 'GENETIC_14Apr2025.csv'
full_df = pd.read_csv(csv_file_path3)
# Updated list of column names, focusing on volumetric measurements
selected_columns = [
'RID', 'APVOLUME'
]
# Create a dictionary of DataFrames, each containing only one of th
dataframe3 = pd.read_csv(csv_file_path3, usecols=selected_columns)
# This will create DataFrames for the columns that exist in the CSV
# preventing KeyError in case some columns are missing
print(dataframe3.head())
```

```
   RID  APVOLUME
0    2      10.0
1    3      10.0
2    4       8.0
3    5      10.0
4    7       9.0
```

In [ ]:
```
csv_file_path4 = '/content/UPENNBIOMK_ROCHE_ELECSYS_14Apr2025.csv'

full_df = pd.read_csv(csv_file_path4)
# Updated list of column names, focusing on volumetric measurements
selected_columns = [
'RID', 'ABETA42','TAU','PTAU'
]
# Create a dictionary of DataFrames, each containing only one of th
dataframe4 = pd.read_csv(csv_file_path4, usecols=selected_columns)
# This will create DataFrames for the columns that exist in the CSV
# preventing KeyError in case some columns are missing
print(dataframe4.head())
```

```
   RID  ABETA42    TAU   PTAU
0    3    741.5  239.7  22.83
1    3    601.4  251.7  24.18
2    4   1501.0  153.1  13.29
3    4   1176.0  159.7  13.30
4    5    547.3  337.0  33.43
```

In [ ]:
```
import pandas as pd

csv_file_path5 = 'UGOTPTAU181_06_18_20_14Apr2025 (1).csv'

full_df = pd.read_csv(csv_file_path4)
# Updated list of column names, focusing on volumetric measurements
selected_columns = [
'RID','PLASMAPTAU181',
```

```
]
# Create a dictionary of DataFrames, each containing only one of th
dataframe5 = pd.read_csv(csv_file_path5, usecols=selected_columns)
# This will create DataFrames for the columns that exist in the CSV
# preventing KeyError in case some columns are missing
print(dataframe5.head())
```

```
   RID  PLASMAPTAU181
0    2         11.939
1    2         12.936
2    2         13.563
3    2         15.506
4    8         18.305
```

In [ ]:

```
csv_file_path = 'UCSFFSX7_14Apr2025.csv'

full_df = pd.read_csv(csv_file_path)
# Updated list of column names, focusing on volumetric measurements
"""selected_columns = [
"ST58CV", "ST58SA", "ST58TA", "ST58TS", # Left Superior Temporal
"ST117CV", "ST117SA", "ST117TA", "ST117TS", # Right Superior Tempor
"ST40CV", "ST40SA", "ST40TA", "ST40TS", # Left Middle Temporal
"ST99CV", "ST99SA", "ST99TA", "ST99TS", # Right Middle Temporal
"ST32CV", "ST32SA", "ST32TA", "ST32TS", # Left Inferior Temporal
"ST91CV", "ST91SA", "ST91TA", "ST91TS", # Right Inferior Temporal
"ST60CV", "ST60SA", "ST60TA", "ST60TS", # Left Temporal Pole
"ST119CV", "ST119SA", "ST119TA", "ST119TS", # Right Temporal Pole
"ST62CV", "ST62SA", "ST62TA", "ST62TS", # Left Transverse Temporal
"ST121CV", "ST121SA", "ST121TA", "ST121TS" # Right Transverse Tempo
]"""
selected_columns = [
"RID",
"ST58TA", # Cortical Thickness Average of Left Superior Temporal
"ST117TA", # Cortical Thickness Average of Right Superior Temporal
"ST40TA", # Cortical Thickness Average of Left Middle Temporal
"ST99TA", # Cortical Thickness Average of Right Middle Temporal
"ST32TA", # Cortical Thickness Average of Left Inferior Temporal
"ST91TA", # Cortical Thickness Average of Right Inferior Temporal
"ST60TA", # Cortical Thickness Average of Left Temporal Pole
"ST119TA", # Cortical Thickness Average of Right Temporal Pole
"ST62TA", # Cortical Thickness Average of Left Transverse Temporal
"ST121TA" # Cortical Thickness Average of Right Transverse Temporal
]
# Create a dictionary of DataFrames, each containing only one of th
dataframe9 = pd.read_csv(csv_file_path, usecols=selected_columns)
# This will create DataFrames for the columns that exist in the CSV
# preventing KeyError in case some columns are missing
print(dataframe9.head())
```

```
    RID  ST117TA  ST119TA  ST121TA  ST32TA  ST40TA  ST58TA  ST60TA
ST62TA  \
0  4213    2.349    3.802    2.145   2.703   2.568   2.471   3.568
2.095
1  4453    2.571    3.739    2.360   2.509   2.560   2.596   3.785
2.348
2  4104    2.912    3.865    2.311   2.706   2.663   2.756   3.508
2.421
3  2153    2.954    3.855    2.770   2.758   2.780   2.922   3.402
2.448
4  4303    2.554    3.347    2.266   2.693   2.754   2.644   3.303
2.233

   ST91TA  ST99TA
0   2.463   2.498
1   2.734   2.651
2   2.862   2.903
3   2.641   2.801
4   2.824   2.774
```

In [ ]:
```python
csv_file_path = 'PTDEMOG_14Apr2025 (1).csv'

full_df = pd.read_csv(csv_file_path)
# Updated list of column names, focusing on volumetric measurements
selected_columns = [
"RID",
# Demographic and Background Information
"PTGENDER", "PTDOB",
# Clinical and Site-Specific Data
#"PTDOBYY", "PTMARRY", "PTEDUCAT","PTETHCAT",
]
# Create a dictionary of DataFrames, each containing only one of th
dataframe7 = pd.read_csv(csv_file_path, usecols=selected_columns)
# This will create DataFrames for the columns that exist in the CSV
# preventing KeyError in case some columns are missing
print(dataframe7.head())
```

```
   RID  PTGENDER    PTDOB
0    2       1.0  04/1931
1    1       2.0  12/1944
2    3       1.0  05/1924
3    4       1.0  01/1938
4    5       1.0  12/1931
```

In [ ]:
```python
csv_file_path = 'MEDHIST_14Apr2025.csv'

full_df = pd.read_csv(csv_file_path)
# Updated list of column names, focusing on volumetric measurements
selected_columns = [
"RID",
# Medical Conditions Related to Brain Structure
"MH14ALCH", # Alcohol Abuse
"MH15DRUG", # Drug Abuse
"MH16SMOK", # Smoking
"MH2NEURL", # Neurologic (other than AD)
"MHPSYCH" # Psychiatric Conditions
]
```

```
# Create a dictionary of DataFrames, each containing only one of th
dataframe8 = pd.read_csv(csv_file_path, usecols=selected_columns)
# This will create DataFrames for the columns that exist in the CSV
# preventing KeyError in case some columns are missing
print(dataframe8.head())
```

```
   RID  MHPSYCH  MH2NEURL  MH14ALCH  MH15DRUG  MH16SMOK
0    2        0         0         0         0         0
1    1        0         0         0         0         0
2    3        0         0         0         0         1
3    4        0         0         0         0         1
4    5        0         0         0         0         1
```

In [ ]:
```
dfs = [dataframe3, dataframe4, dataframe5, dataframe7, dataframe8,

# Check if 'RID' is present in each DataFrame
rid_check = [True if 'RID' in df.columns else False for df in dfs]

# Display all columns when printing DataFrames
pd.set_option('display.max_columns', None)

# Print the check results
print("RID presence in each DataFrame:", rid_check)

# Merge all DataFrames on 'RID' using inner join
merged_df = reduce(lambda left, right: pd.merge(left, right, on='RI

# Display the merged DataFrame
print(merged_df.head())
```

```
RID presence in each DataFrame: [True, True, True, True, True, True]
   RID  APVOLUME  ABETA42    TAU   PTAU  PLASMAPTAU181  PTGENDER
PTDOB  \
0   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
1   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
2   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
3   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
4   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928

   MHPSYCH  MH2NEURL  MH14ALCH  MH15DRUG  MH16SMOK  ST117TA  ST119TA
ST121TA  \
0        0         0         0         0         0    2.318    3.495
2.080
1        0         0         0         0         0    2.412    2.758
2.469
2        0         0         0         0         0    2.318    3.495
2.080
3        0         0         0         0         0    2.412    2.758
2.469
4        0         0         0         0         0    2.318    3.495
2.080

   ST32TA  ST40TA  ST58TA  ST60TA  ST62TA  ST91TA  ST99TA
0   2.767   2.740   2.304   3.670   1.718   2.528   2.463
1   2.744   2.738   2.548   3.687   2.177   2.229   2.384
2   2.767   2.740   2.304   3.670   1.718   2.528   2.463
3   2.744   2.738   2.548   3.687   2.177   2.229   2.384
4   2.767   2.740   2.304   3.670   1.718   2.528   2.463
```

In [ ]:

```
pd.set_option('display.max_columns', None)
print(merged_df.head())
```

```
   RID  APVOLUME  ABETA42    TAU   PTAU  PLASMAPTAU181  PTGENDER
PTDOB  \
0   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
1   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
2   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
3   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928
4   31      10.0   1774.0  266.8  22.55         18.642       2.0  0
1/1928

   MHPSYCH  MH2NEURL  MH14ALCH  MH15DRUG  MH16SMOK  ST117TA  ST119TA
ST121TA  \
0        0         0         0         0         0    2.318    3.495
2.080
1        0         0         0         0         0    2.412    2.758
2.469
2        0         0         0         0         0    2.318    3.495
2.080
3        0         0         0         0         0    2.412    2.758
2.469
4        0         0         0         0         0    2.318    3.495
2.080

   ST32TA  ST40TA  ST58TA  ST60TA  ST62TA  ST91TA  ST99TA
0   2.767   2.740   2.304   3.670   1.718   2.528   2.463
1   2.744   2.738   2.548   3.687   2.177   2.229   2.384
2   2.767   2.740   2.304   3.670   1.718   2.528   2.463
3   2.744   2.738   2.548   3.687   2.177   2.229   2.384
4   2.767   2.740   2.304   3.670   1.718   2.528   2.463
```

In [ ]:
```
print(merged_df.describe())
```

```
                 RID       APVOLUME        ABETA42            TAU  \
count  742688.000000  579314.000000  740998.000000  740588.000000
mean     2545.892749       0.444097    1238.469569     271.269763
std      1516.797451       6.235802     637.193384     132.301563
min        23.000000      -4.000000     203.000000      80.080000
25%      1261.000000      -4.000000     702.400000     186.400000
50%      2245.000000      -4.000000    1106.000000     238.800000
75%      4175.000000       9.000000    1689.000000     310.000000
max      5296.000000      12.000000    3949.000000    1018.000000

                PTAU  PLASMAPTAU181       PTGENDER        MHPSYCH  \
count  737202.000000  742688.000000  703651.000000  742688.000000
mean       25.252673      16.011811       1.498388       0.405950
std        14.873441       9.512347       0.640717       0.491075
min         8.260000       0.468000      -4.000000       0.000000
25%        16.300000      10.300000       1.000000       0.000000
50%        21.090000      14.564000       2.000000       0.000000
75%        28.580000      19.264000       2.000000       1.000000
max       103.700000     451.398000       2.000000       1.000000

            MH2NEURL       MH14ALCH       MH15DRUG       MH16SMOK  \
```

```
count  742688.000000  742688.000000  742688.000000  742688.000000
mean        0.386790       0.029850       0.011048       0.402071
std         0.487015       0.170173       0.104526       0.490316
min         0.000000       0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000       0.000000
75%         1.000000       0.000000       0.000000       1.000000
max         1.000000       1.000000       1.000000       1.000000

              ST117TA        ST119TA        ST121TA         ST32TA  \
count  736958.000000  736958.000000  742128.000000  736958.000000
mean        2.591584       3.453158       2.305323       2.645257
std         0.197361       0.374494       0.218933       0.181849
min         1.644000       1.407000       1.413000       1.729000
25%         2.469000       3.295000       2.159000       2.540000
50%         2.601000       3.517000       2.310000       2.653000
75%         2.710000       3.719750       2.450000       2.767000
max         3.124000       4.315000       2.970000       3.197000

               ST40TA         ST58TA         ST60TA         ST62TA  \
count  736958.000000  736958.000000  736958.000000  742128.000000
mean        2.659840       2.566967       3.373313       2.273667
std         0.182294       0.195884       0.372920       0.206554
min         1.773000       1.723000       1.423000       1.396000
25%         2.563000       2.457000       3.190000       2.148000
50%         2.673000       2.583000       3.424000       2.278000
75%         2.782000       2.702000       3.606000       2.413000
max         3.147000       3.010000       4.310000       2.884000

               ST91TA         ST99TA
count  736958.000000  736958.000000
mean        2.669802       2.692279
std         0.176935       0.180707
min         1.747000       1.661000
25%         2.573000       2.598000
50%         2.694000       2.703000
75%         2.805000       2.802000
max         3.195000       3.168000
```

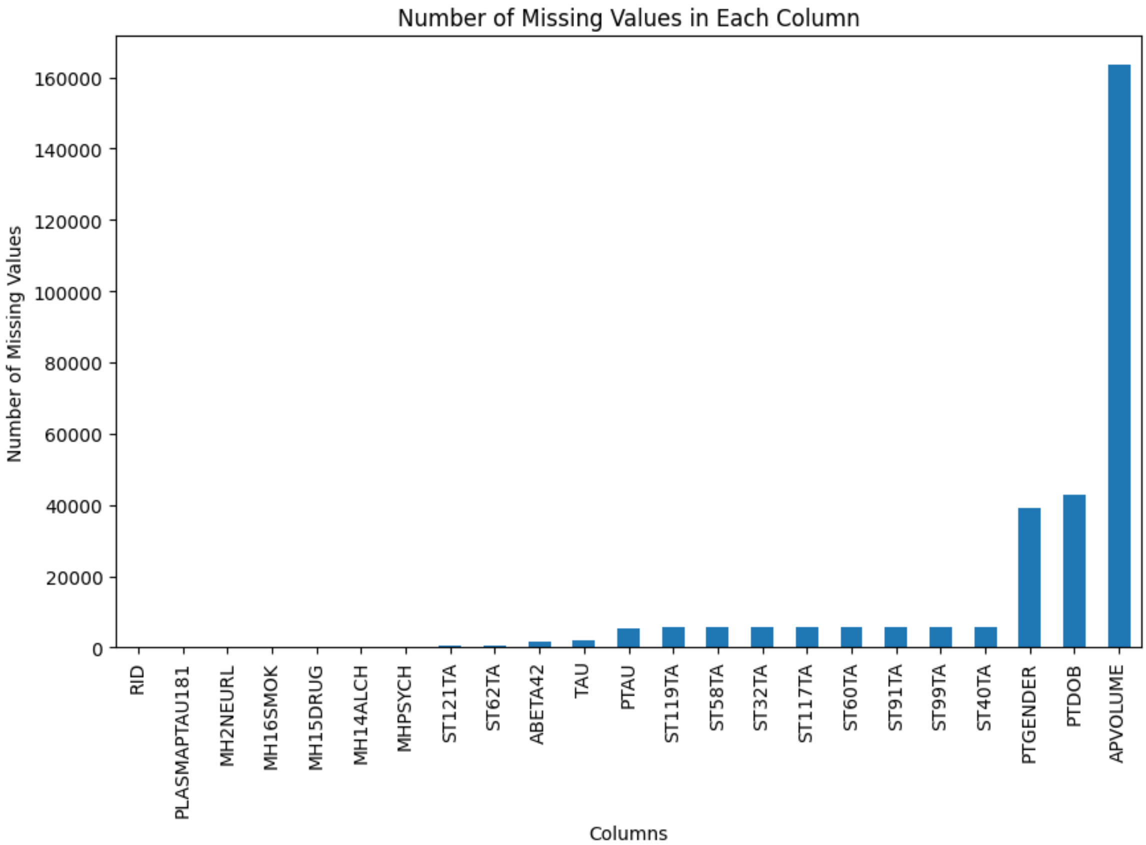
In [ ]:

```
print(merged_df.isnull().sum())
```

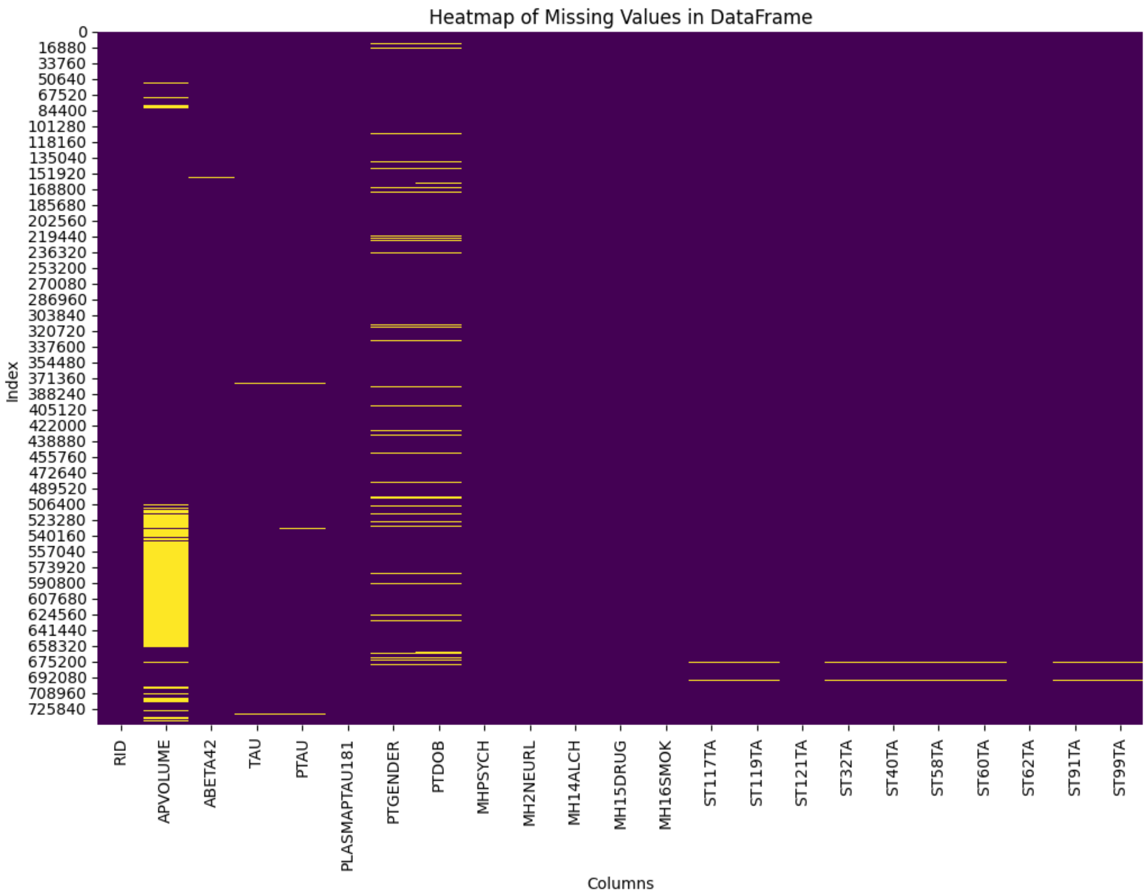
```
RID                  0
APVOLUME        163374
ABETA42           1690
TAU               2100
PTAU              5486
PLASMAPTAU181        0
PTGENDER         39037
PTDOB            42802
MHPSYCH              0
MH2NEURL             0
MH14ALCH             0
MH15DRUG             0
MH16SMOK             0
ST117TA           5730
ST119TA           5730
ST121TA            560
ST32TA            5730
ST40TA            5730
ST58TA            5730
ST60TA            5730
ST62TA             560
ST91TA            5730
ST99TA            5730
dtype: int64
```

In [ ]:

```
missing_values = merged_df.isnull().sum()
missing_values.sort_values(inplace=True) # Sort the values
missing_values.plot(kind='bar', figsize=(10, 6)) # Create a bar plo
plt.xlabel('Columns')
plt.ylabel('Number of Missing Values')
plt.title('Number of Missing Values in Each Column')
plt.show()
```



In [ ]:
```python
plt.figure(figsize=(12, 8))
sns.heatmap(merged_df.isnull(), cbar=False, cmap='viridis')
plt.title('Heatmap of Missing Values in DataFrame')
plt.xlabel('Columns')
plt.ylabel('Index')
plt.show()
```



In [ ]:
```python
cleaned_df_dropna = merged_df.dropna()
print(cleaned_df_dropna.describe())
```

```
                 RID       APVOLUME        ABETA42            TAU  \
count  537865.000000  537865.000000  537865.000000  537865.000000
mean     2530.891097       0.390705    1261.005796     266.261582
std      1533.006742       6.215089     643.410172     115.849659
min        23.000000      -4.000000     203.000000      88.690000
25%      1261.000000      -4.000000     717.200000     186.600000
50%      2245.000000      -4.000000    1146.000000     239.100000
75%      4187.000000       9.000000    1718.000000     313.200000
max      5296.000000      12.000000    3949.000000     915.800000

                PTAU  PLASMAPTAU181       PTGENDER        MHPSYCH  \
count  537865.000000  537865.000000  537865.000000  537865.000000
mean       24.608651      16.228029       1.512430       0.400106
std        12.965423       9.808985       0.499846       0.489920
min         8.260000       0.468000       1.000000       0.000000
25%        16.400000      10.274000       1.000000       0.000000
50%        21.090000      14.533000       2.000000       0.000000
75%        28.710000      19.988000       2.000000       1.000000
max       103.700000     451.398000       2.000000       1.000000
```

```
          MH2NEURL       MH14ALCH       MH15DRUG       MH16SMOK  \
count  537865.000000  537865.000000  537865.000000  537865.000000
mean        0.378498       0.028171       0.012076       0.425254
std         0.485013       0.165460       0.109223       0.494382
min         0.000000       0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000       0.000000
75%         1.000000       0.000000       0.000000       1.000000
max         1.000000       1.000000       1.000000       1.000000

             ST117TA        ST119TA        ST121TA         ST32TA  \
count  537865.000000  537865.000000  537865.000000  537865.000000
mean        2.582546       3.431361       2.303026       2.637437
std         0.203246       0.384000       0.216833       0.185909
min         1.644000       1.407000       1.413000       1.729000
25%         2.457000       3.274000       2.161000       2.532000
50%         2.593000       3.503000       2.309000       2.648000
75%         2.704000       3.692000       2.443000       2.762000
max         3.124000       4.315000       2.970000       3.197000

              ST40TA         ST58TA         ST60TA         ST62TA  \
count  537865.000000  537865.000000  537865.000000  537865.000000
mean        2.652819       2.560486       3.347277       2.270687
std         0.188352       0.201863       0.378968       0.211328
min         1.773000       1.723000       1.423000       1.396000
25%         2.553000       2.446000       3.164000       2.141000
50%         2.667000       2.578000       3.411000       2.275000
75%         2.780000       2.701000       3.581000       2.413000
max         3.147000       3.010000       4.310000       2.884000

              ST91TA         ST99TA
count  537865.000000  537865.000000
mean        2.663668       2.680271
std         0.182223       0.182187
min         1.747000       1.661000
25%         2.565000       2.586000
50%         2.688000       2.692000
75%         2.800000       2.796000
max         3.195000       3.168000
```

In [ ]:
```
cleaned_df_mean = merged_df.copy()

for column in cleaned_df_mean.select_dtypes(include=['float64', 'in
    cleaned_df_mean[column].fillna(cleaned_df_mean[column].mean(),
```

```
<ipython-input-19-eafc50d4d64f>:4: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained assignm
ent using an inplace method.
The behavior will change in pandas 3.0. This inplace method will nev
er work because the intermediate object on which we are setting valu
es always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try u
sing 'df.method({col: value}, inplace=True)' or df[col] = df[col].me
thod(value) instead, to perform the operation inplace on the origina
l object.


  cleaned_df_mean[column].fillna(cleaned_df_mean[column].mean(), inp
lace=True)
```

In [ ]:
```
def parse_date(date_str):
    try:
        # Try parsing as "4/1/1931" format
        return pd.to_datetime(date_str, format='%m/%d/%Y')
    except ValueError:
        try:
            # Try parsing as "Apr-31" format
            month, year = date_str.split('-')
            month_num = pd.to_datetime(month, format='%b').month
            return pd.to_datetime(f'1-{month_num}-{year}', format='
        except ValueError:
            try:
                # Try parsing as "01/1934" format
                month, year = date_str.split('/')
                return pd.to_datetime(f'1-{month}-{year}', format='
            except ValueError:
                # Return NaT for anything else
                return pd.NaT

# Apply it
cleaned_df_dropna['PTDOB'] = cleaned_df_dropna['PTDOB'].apply(parse_

# Print a few values to confirm
print(cleaned_df_dropna['PTDOB'].head())
```

```
0   1928-01-01
1   1928-01-01
2   1928-01-01
3   1928-01-01
4   1928-01-01
Name: PTDOB, dtype: datetime64[ns]
```

```
<ipython-input-20-a20a3a60c4fb>:21: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
  cleaned_df_dropna['PTDOB'] = cleaned_df_dropna['PTDOB'].apply(pars
e_date)
```

In [ ]:
```
target_names = [
"ST58TA", # Cortical Thickness Average of Left Superior Temporal
"ST117TA", # Cortical Thickness Average of Right Superior Temporal
"ST40TA", # Cortical Thickness Average of Left Middle Temporal
"ST99TA", # Cortical Thickness Average of Right Middle Temporal
"ST32TA", # Cortical Thickness Average of Left Inferior Temporal
"ST91TA", # Cortical Thickness Average of Right Inferior Temporal
"ST60TA", # Cortical Thickness Average of Left Temporal Pole
"ST119TA", # Cortical Thickness Average of Right Temporal Pole
"ST62TA", # Cortical Thickness Average of Left Transverse Temporal
"ST121TA" # Cortical Thickness Average of Right Transverse Temporal
]
# Features DataFrame (X)
# Drop target columns from the main DataFrame to create the feature
X = cleaned_df_dropna.drop(columns=target_names + ['RID'])
# Targets DataFrame (y)
# Select only the target columns for the target DataFrame
y = cleaned_df_dropna[target_names]
```

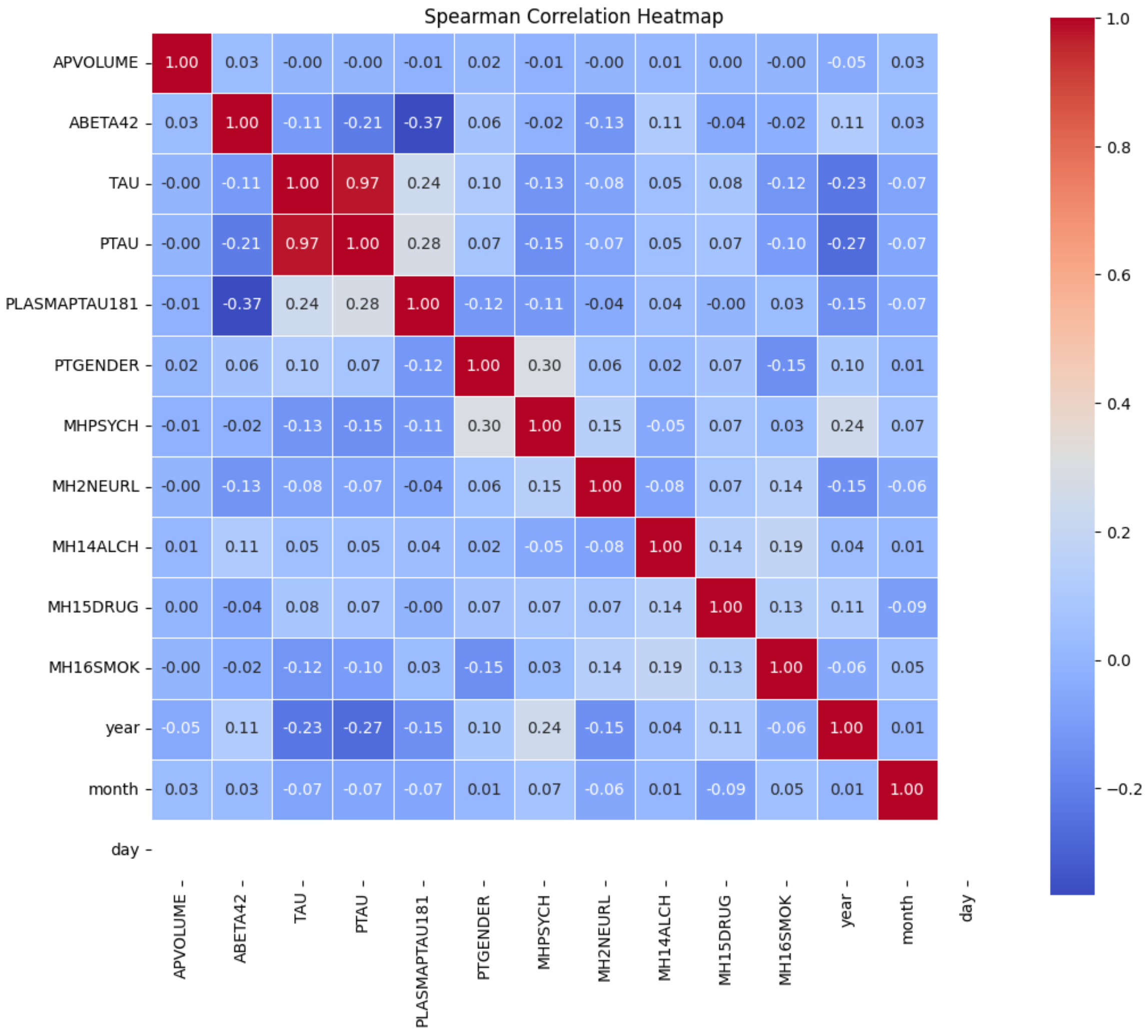
In [ ]:
```
X['PTDOB'] = pd.to_datetime(X['PTDOB'])
X['year'] = X['PTDOB'].dt.year
X['month'] = X['PTDOB'].dt.month
X['day'] = X['PTDOB'].dt.day
# Now, you can drop the original datetime column if needed
X = X.drop(columns=['PTDOB'])
```

In [ ]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

# Columns to use for outlier detection and scaling
cols = ['APVOLUME', 'ABETA42', 'TAU', 'PTAU', 'PLASMAPTAU181']

# Calculate Q1, Q3, and IQR
Q1 = X[cols].quantile(0.25)
Q3 = X[cols].quantile(0.75)
IQR = Q3 - Q1

# Define an outlier mask (True = not an outlier)
outlier_mask = ~((X[cols] < (Q1 - 1.5 * IQR)) | (X[cols] > (Q3 + 1.

# Apply the mask to filter out outliers
X_clean = X.loc[outlier_mask, cols]
y_clean = y[outlier_mask]

# Normalize the data using Min-Max Scaling
scaler = MinMaxScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X_clean), columns=cols

# Display shapes before and after cleaning
print("Original shape of X:", X.shape)
print("New shape of X after outlier removal and scaling:", X_scaled
print("Original shape of y:", y.shape)
print("New shape of y after outlier and scaling process:", y_clean.
```

```
Original shape of X: (537865, 14)
New shape of X after outlier removal and scaling: (484436, 5)
Original shape of y: (537865, 10)
New shape of y after outlier and scaling process: (484436, 10)
```

In [ ]:
```
correlation_matrix = X.corr(method='spearman')
```

In [ ]:
```
plt.figure(figsize=(12, 10)) # Set the figure size as needed
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwa
cbar=True, square=True, linewidths=.5)
plt.title('Spearman Correlation Heatmap')
plt.show()
```



In [ ]:
```
cleaned_df_dropna.shape
```

Out[ ]: (537865, 23)

In [ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np

# Split the dataset into training and testing sets
```

```
X_train_full, X_test, y_train_full, y_test = train_test_split(X, y,

# Now split the training set into four parts, simulating distributi
parts_X = np.array_split(X_train_full, 4)
parts_y = np.array_split(y_train_full, 4)

# Define the models to be used
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'k-NN': KNeighborsRegressor(n_neighbors=10)
}

# Train models on each client's data for each target and store pred
client_predictions = {
    name: {col: [] for col in y_train_full.columns}
    for name in models.keys()
}

for i in range(4):
    X_train, y_train = parts_X[i], parts_y[i]
    for name, model in models.items():
        for col in y_train.columns:
            model.fit(X_train, y_train[col])
            predictions = model.predict(X_test)
            client_predictions[name][col].append(predictions)

# Average predictions from each client for each target variable and
for name, targets in client_predictions.items():
    print(f"\nResults for {name}:")
    for target_col, predictions in targets.items():
        average_prediction = np.mean(predictions, axis=0)
        mse = mean_squared_error(y_test[target_col], average_predic
        r2 = r2_score(y_test[target_col], average_prediction)
        print(f"  {target_col} - Averaged MSE: {mse:.4f}, Averaged
```

```
/usr/local/lib/python3.11/dist-packages/numpy/_core/fromnumeric.py:5
7: FutureWarning: 'DataFrame.swapaxes' is deprecated and will be rem
oved in a future version. Please use 'DataFrame.transpose' instead.
  return bound(*args, **kwds)
```

```
Results for Linear Regression:
  ST58TA - Averaged MSE: 0.0287, Averaged R^2 Score: 0.2937
  ST117TA - Averaged MSE: 0.0287, Averaged R^2 Score: 0.3011
  ST40TA - Averaged MSE: 0.0287, Averaged R^2 Score: 0.1904
  ST99TA - Averaged MSE: 0.0277, Averaged R^2 Score: 0.1649
  ST32TA - Averaged MSE: 0.0302, Averaged R^2 Score: 0.1293
  ST91TA - Averaged MSE: 0.0282, Averaged R^2 Score: 0.1500
  ST60TA - Averaged MSE: 0.1342, Averaged R^2 Score: 0.0553
  ST119TA - Averaged MSE: 0.1285, Averaged R^2 Score: 0.1191
  ST62TA - Averaged MSE: 0.0389, Averaged R^2 Score: 0.1289
  ST121TA - Averaged MSE: 0.0398, Averaged R^2 Score: 0.1548

Results for Decision Tree:
  ST58TA - Averaged MSE: 0.0042, Averaged R^2 Score: 0.8959
  ST117TA - Averaged MSE: 0.0038, Averaged R^2 Score: 0.9087
  ST40TA - Averaged MSE: 0.0043, Averaged R^2 Score: 0.8793
  ST99TA - Averaged MSE: 0.0039, Averaged R^2 Score: 0.8820
  ST32TA - Averaged MSE: 0.0053, Averaged R^2 Score: 0.8474
  ST91TA - Averaged MSE: 0.0059, Averaged R^2 Score: 0.8217
  ST60TA - Averaged MSE: 0.0276, Averaged R^2 Score: 0.8055
  ST119TA - Averaged MSE: 0.0324, Averaged R^2 Score: 0.7778
  ST62TA - Averaged MSE: 0.0067, Averaged R^2 Score: 0.8497
  ST121TA - Averaged MSE: 0.0081, Averaged R^2 Score: 0.8288

Results for Random Forest:
  ST58TA - Averaged MSE: 0.0042, Averaged R^2 Score: 0.8969
  ST117TA - Averaged MSE: 0.0037, Averaged R^2 Score: 0.9096
  ST40TA - Averaged MSE: 0.0042, Averaged R^2 Score: 0.8805
  ST99TA - Averaged MSE: 0.0039, Averaged R^2 Score: 0.8831
  ST32TA - Averaged MSE: 0.0052, Averaged R^2 Score: 0.8491
  ST91TA - Averaged MSE: 0.0059, Averaged R^2 Score: 0.8235
  ST60TA - Averaged MSE: 0.0273, Averaged R^2 Score: 0.8078
  ST119TA - Averaged MSE: 0.0321, Averaged R^2 Score: 0.7799
  ST62TA - Averaged MSE: 0.0067, Averaged R^2 Score: 0.8512
  ST121TA - Averaged MSE: 0.0080, Averaged R^2 Score: 0.8304

Results for k-NN:
  ST58TA - Averaged MSE: 0.0045, Averaged R^2 Score: 0.8884
  ST117TA - Averaged MSE: 0.0040, Averaged R^2 Score: 0.9021
  ST40TA - Averaged MSE: 0.0045, Averaged R^2 Score: 0.8724
  ST99TA - Averaged MSE: 0.0041, Averaged R^2 Score: 0.8758
  ST32TA - Averaged MSE: 0.0055, Averaged R^2 Score: 0.8413
  ST91TA - Averaged MSE: 0.0061, Averaged R^2 Score: 0.8162
  ST60TA - Averaged MSE: 0.0284, Averaged R^2 Score: 0.7998
  ST119TA - Averaged MSE: 0.0335, Averaged R^2 Score: 0.7707
  ST62TA - Averaged MSE: 0.0070, Averaged R^2 Score: 0.8427
  ST121TA - Averaged MSE: 0.0084, Averaged R^2 Score: 0.8216
```

In [ ]:

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

```
# Assuming 'X' and 'y' are defined (features and targets)
# Replace with actual dataset loading if needed
# Example:
# data = pd.read_csv("cleaned_data.csv")
# X = data.drop(columns=['target1', 'target2'])  # replace with act
# y = data[['target1', 'target2']]

# Split the entire dataset into a training set and a testing set
X_train_full, X_test, y_train_full, y_test = train_test_split(X, y,

# Now split the training set into four parts, simulating distributi
parts_X = np.array_split(X_train_full, 4)
parts_y = np.array_split(y_train_full, 4)

models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'k-NN': KNeighborsRegressor(n_neighbors=10)
}

# Train models on each client's data for each target and store pred
client_predictions = {name: {col: [] for col in y_train_full.column
weights = [len(part) for part in parts_y]  # Number of samples in e

for i in range(4):
    X_train, y_train = parts_X[i], parts_y[i]
    for name, model in models.items():
        for col in y_train.columns:
            model.fit(X_train, y_train[col])
            predictions = model.predict(X_test)
            client_predictions[name][col].append(predictions)

# Calculate weighted average of predictions from each client for ea
for name, targets in client_predictions.items():
    print(f"Results for {name}:")
    for target_col, predictions in targets.items():
        # Compute weighted average
        weighted_average_prediction = np.average(predictions, axis=
        mse = mean_squared_error(y_test[target_col], weighted_avera
        r2 = r2_score(y_test[target_col], weighted_average_predicti
        print(f"  {target_col} - Weighted Averaged MSE: {mse:.4f}, '
```

```
/usr/local/lib/python3.11/dist-packages/numpy/_core/fromnumeric.py:5
7: FutureWarning: 'DataFrame.swapaxes' is deprecated and will be rem
oved in a future version. Please use 'DataFrame.transpose' instead.
  return bound(*args, **kwds)
Results for Linear Regression:
  ST58TA - Weighted Averaged MSE: 0.0287, Weighted Averaged R2: 0.29
37
  ST117TA - Weighted Averaged MSE: 0.0287, Weighted Averaged R2: 0.3
011
  ST40TA - Weighted Averaged MSE: 0.0287, Weighted Averaged R2: 0.19
04
  ST99TA - Weighted Averaged MSE: 0.0277, Weighted Averaged R2: 0.16
```

```
49
  ST32TA - Weighted Averaged MSE: 0.0302, Weighted Averaged R2: 0.12
93
  ST91TA - Weighted Averaged MSE: 0.0282, Weighted Averaged R2: 0.15
00
  ST60TA - Weighted Averaged MSE: 0.1342, Weighted Averaged R2: 0.05
53
  ST119TA - Weighted Averaged MSE: 0.1285, Weighted Averaged R2: 0.1
191
  ST62TA - Weighted Averaged MSE: 0.0389, Weighted Averaged R2: 0.12
89
  ST121TA - Weighted Averaged MSE: 0.0398, Weighted Averaged R2: 0.1
548
Results for Decision Tree:
  ST58TA - Weighted Averaged MSE: 0.0042, Weighted Averaged R2: 0.89
59
  ST117TA - Weighted Averaged MSE: 0.0038, Weighted Averaged R2: 0.9
087
  ST40TA - Weighted Averaged MSE: 0.0043, Weighted Averaged R2: 0.87
92
  ST99TA - Weighted Averaged MSE: 0.0039, Weighted Averaged R2: 0.88
19
  ST32TA - Weighted Averaged MSE: 0.0053, Weighted Averaged R2: 0.84
74
  ST91TA - Weighted Averaged MSE: 0.0059, Weighted Averaged R2: 0.82
17
  ST60TA - Weighted Averaged MSE: 0.0276, Weighted Averaged R2: 0.80
55
  ST119TA - Weighted Averaged MSE: 0.0324, Weighted Averaged R2: 0.7
777
  ST62TA - Weighted Averaged MSE: 0.0067, Weighted Averaged R2: 0.84
97
  ST121TA - Weighted Averaged MSE: 0.0081, Weighted Averaged R2: 0.8
287
Results for Random Forest:
  ST58TA - Weighted Averaged MSE: 0.0042, Weighted Averaged R2: 0.89
69
  ST117TA - Weighted Averaged MSE: 0.0037, Weighted Averaged R2: 0.9
095
  ST40TA - Weighted Averaged MSE: 0.0042, Weighted Averaged R2: 0.88
05
  ST99TA - Weighted Averaged MSE: 0.0039, Weighted Averaged R2: 0.88
31
  ST32TA - Weighted Averaged MSE: 0.0052, Weighted Averaged R2: 0.84
90
  ST91TA - Weighted Averaged MSE: 0.0059, Weighted Averaged R2: 0.82
36
  ST60TA - Weighted Averaged MSE: 0.0273, Weighted Averaged R2: 0.80
78
  ST119TA - Weighted Averaged MSE: 0.0321, Weighted Averaged R2: 0.7
799
  ST62TA - Weighted Averaged MSE: 0.0067, Weighted Averaged R2: 0.85
12
  ST121TA - Weighted Averaged MSE: 0.0080, Weighted Averaged R2: 0.8
304
Results for k-NN:
```

```
  ST58TA - Weighted Averaged MSE: 0.0045, Weighted Averaged R2: 0.88
84
  ST117TA - Weighted Averaged MSE: 0.0040, Weighted Averaged R2: 0.9
021
  ST40TA - Weighted Averaged MSE: 0.0045, Weighted Averaged R2: 0.87
24
  ST99TA - Weighted Averaged MSE: 0.0041, Weighted Averaged R2: 0.87
58
  ST32TA - Weighted Averaged MSE: 0.0055, Weighted Averaged R2: 0.84
13
  ST91TA - Weighted Averaged MSE: 0.0061, Weighted Averaged R2: 0.81
62
  ST60TA - Weighted Averaged MSE: 0.0284, Weighted Averaged R2: 0.79
98
  ST119TA - Weighted Averaged MSE: 0.0335, Weighted Averaged R2: 0.7
707
  ST62TA - Weighted Averaged MSE: 0.0070, Weighted Averaged R2: 0.84
27
  ST121TA - Weighted Averaged MSE: 0.0084, Weighted Averaged R2: 0.8
216
```

In [ ]:

```
!pip install xgboost
```

```
Requirement already satisfied: xgboost in /usr/local/lib/python3.11/
dist-packages (2.1.4)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/di
st-packages (from xgboost) (2.0.2)
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/py
thon3.11/dist-packages (from xgboost) (2.21.5)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/di
st-packages (from xgboost) (1.14.1)
```

In [ ]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
import xgboost as xgb

# Assuming 'X' and 'y' are defined
# X = features, y = DataFrame with multiple target columns

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size

# Initialize XGBoost Regressor
model = xgb.XGBRegressor(objective='reg:squarederror')

# Train and evaluate model for each target variable
for target_col in y_train.columns:
    print(f"Results for target variable: {target_col}")
    model.fit(X_train, y_train[target_col])
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test[target_col], y_pred)
    r2 = r2_score(y_test[target_col], y_pred)
    print(f"XGBoost - MSE: {mse:.4f}, R^2 Score: {r2:.4f}")
    print()
```

```
Results for target variable: ST58TA
XGBoost - MSE: 0.0043, R^2 Score: 0.8933

Results for target variable: ST117TA
XGBoost - MSE: 0.0039, R^2 Score: 0.9057

Results for target variable: ST40TA
XGBoost - MSE: 0.0044, R^2 Score: 0.8761

Results for target variable: ST99TA
XGBoost - MSE: 0.0041, R^2 Score: 0.8776

Results for target variable: ST32TA
XGBoost - MSE: 0.0054, R^2 Score: 0.8450

Results for target variable: ST91TA
XGBoost - MSE: 0.0060, R^2 Score: 0.8204

Results for target variable: ST60TA
XGBoost - MSE: 0.0282, R^2 Score: 0.8018

Results for target variable: ST119TA
XGBoost - MSE: 0.0326, R^2 Score: 0.7769

Results for target variable: ST62TA
XGBoost - MSE: 0.0069, R^2 Score: 0.8456

Results for target variable: ST121TA
XGBoost - MSE: 0.0082, R^2 Score: 0.8256
```

In [ ]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Assuming X and y are already defined DataFrames
# X = features, y = target variables (possibly multi-output)

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size

# Initialize regression models
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'k-NN': KNeighborsRegressor(n_neighbors=10)
}

# Train and evaluate each model for each target variable
for target_col in y_train.columns:
    print(f"\nResults for target variable: {target_col}")
```

```
    for name, model in models.items():
        model.fit(X_train, y_train[target_col])
        y_pred = model.predict(X_test)
        mse = mean_squared_error(y_test[target_col], y_pred)
        r2 = r2_score(y_test[target_col], y_pred)
        accuracy = model.score(X_test, y_test[target_col])  # Same 
        print(f"{name} - MSE: {mse:.4f}, R^2 Score: {r2:.4f}, Accur
```

```
Results for target variable: ST58TA
Linear Regression - MSE: 0.0287, R^2 Score: 0.2937, Accuracy: 0.2937
Decision Tree - MSE: 0.0042, R^2 Score: 0.8959, Accuracy: 0.8959
Random Forest - MSE: 0.0042, R^2 Score: 0.8960, Accuracy: 0.8960
k-NN - MSE: 0.0045, R^2 Score: 0.8891, Accuracy: 0.8891

Results for target variable: ST117TA
Linear Regression - MSE: 0.0287, R^2 Score: 0.3011, Accuracy: 0.3011
Decision Tree - MSE: 0.0037, R^2 Score: 0.9090, Accuracy: 0.9090
Random Forest - MSE: 0.0037, R^2 Score: 0.9090, Accuracy: 0.9090
k-NN - MSE: 0.0040, R^2 Score: 0.9031, Accuracy: 0.9031

Results for target variable: ST40TA
Linear Regression - MSE: 0.0287, R^2 Score: 0.1904, Accuracy: 0.1904
Decision Tree - MSE: 0.0043, R^2 Score: 0.8793, Accuracy: 0.8793
Random Forest - MSE: 0.0043, R^2 Score: 0.8795, Accuracy: 0.8795
k-NN - MSE: 0.0045, R^2 Score: 0.8716, Accuracy: 0.8716

Results for target variable: ST99TA
Linear Regression - MSE: 0.0277, R^2 Score: 0.1649, Accuracy: 0.1649
Decision Tree - MSE: 0.0039, R^2 Score: 0.8823, Accuracy: 0.8823
Random Forest - MSE: 0.0039, R^2 Score: 0.8824, Accuracy: 0.8824
k-NN - MSE: 0.0042, R^2 Score: 0.8744, Accuracy: 0.8744

Results for target variable: ST32TA
Linear Regression - MSE: 0.0302, R^2 Score: 0.1293, Accuracy: 0.1293
Decision Tree - MSE: 0.0053, R^2 Score: 0.8476, Accuracy: 0.8476
Random Forest - MSE: 0.0053, R^2 Score: 0.8478, Accuracy: 0.8478
k-NN - MSE: 0.0056, R^2 Score: 0.8386, Accuracy: 0.8386

Results for target variable: ST91TA
Linear Regression - MSE: 0.0282, R^2 Score: 0.1500, Accuracy: 0.1500
Decision Tree - MSE: 0.0059, R^2 Score: 0.8221, Accuracy: 0.8221
Random Forest - MSE: 0.0059, R^2 Score: 0.8222, Accuracy: 0.8222
k-NN - MSE: 0.0063, R^2 Score: 0.8104, Accuracy: 0.8104

Results for target variable: ST60TA
Linear Regression - MSE: 0.1342, R^2 Score: 0.0553, Accuracy: 0.0553
Decision Tree - MSE: 0.0277, R^2 Score: 0.8054, Accuracy: 0.8054
Random Forest - MSE: 0.0276, R^2 Score: 0.8057, Accuracy: 0.8057
k-NN - MSE: 0.0293, R^2 Score: 0.7941, Accuracy: 0.7941

Results for target variable: ST119TA
Linear Regression - MSE: 0.1285, R^2 Score: 0.1191, Accuracy: 0.1191
Decision Tree - MSE: 0.0325, R^2 Score: 0.7773, Accuracy: 0.7773
Random Forest - MSE: 0.0324, R^2 Score: 0.7776, Accuracy: 0.7776
k-NN - MSE: 0.0343, R^2 Score: 0.7648, Accuracy: 0.7648
```

```
Results for target variable: ST62TA
Linear Regression - MSE: 0.0389, R^2 Score: 0.1289, Accuracy: 0.1289
Decision Tree - MSE: 0.0067, R^2 Score: 0.8496, Accuracy: 0.8496
Random Forest - MSE: 0.0067, R^2 Score: 0.8499, Accuracy: 0.8499
k-NN - MSE: 0.0071, R^2 Score: 0.8406, Accuracy: 0.8406

Results for target variable: ST121TA
Linear Regression - MSE: 0.0398, R^2 Score: 0.1548, Accuracy: 0.1548
Decision Tree - MSE: 0.0081, R^2 Score: 0.8283, Accuracy: 0.8283
Random Forest - MSE: 0.0081, R^2 Score: 0.8286, Accuracy: 0.8286
k-NN - MSE: 0.0086, R^2 Score: 0.8179, Accuracy: 0.8179
```

In [ ]:

```python
import pandas as pd
import numpy as np
import time
from sklearn.model_selection import train_test_split, KFold
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostin
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, r2_score, mean_abso
import xgboost as xgb
from catboost import CatBoostRegressor
import lightgbm as lgb
import matplotlib.pyplot as plt

# Assuming X and y are already defined as your feature and target D
X_train_full, X_test, y_train_full, y_test = train_test_split(X, y,

# Define regression models
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'k-NN': KNeighborsRegressor(n_neighbors=10),
    'Gradient Boosting': GradientBoostingRegressor(),
    'XGBoost': xgb.XGBRegressor(objective='reg:squarederror'),
    'CatBoost': CatBoostRegressor(verbose=0),
    'LightGBM': lgb.LGBMRegressor()
}

# Simulate federated learning with 4 clients
parts_X = np.array_split(X_train_full, 4)
parts_y = np.array_split(y_train_full, 4)

results = []
kf = KFold(n_splits=5, shuffle=True, random_state=42)

# Collect predictions from each client
client_predictions = {
    name: {col: [] for col in y_train_full.columns} for name in mod
}

# Federated-style training
for i in range(4):
    X_train, y_train = parts_X[i], parts_y[i]
```

```
    for name, model in models.items():
        for col in y_train.columns:
            model.fit(X_train, y_train[col])
            predictions = model.predict(X_test)
            client_predictions[name][col].append(predictions)

# Average predictions and compute metrics
for name, targets in client_predictions.items():
    print(f"Results for {name}:")
    for target_col, predictions in targets.items():
        average_prediction = np.mean(predictions, axis=0)
        mse = mean_squared_error(y_test[target_col], average_predic
        mae = mean_absolute_error(y_test[target_col], average_predi
        rmse = np.sqrt(mse)
        mape = np.mean(np.abs((y_test[target_col] - average_predict
        r2 = r2_score(y_test[target_col], average_prediction)
        rse = np.sqrt(np.sum((y_test[target_col] - average_predicti

        print(f" {target_col} - Averaged MSE: {mse:.4f}, MAE: {mae:

        results.append({
            'model': name,
            'target_col': target_col,
            'mse': mse,
            'mae': mae,
            'rmse': rmse,
            'mape': mape,
            'r2': r2,
            'rse': rse
        })

# Classical centralized training
for target_col in y_train_full.columns:
    print(f"\nResults for target variable: {target_col}")
    for name, model in models.items():
        start_time = time.time()
        model.fit(X_train_full, y_train_full[target_col])
        y_pred = model.predict(X_test)
        end_time = time.time()
        elapsed_time = end_time - start_time

        mse = mean_squared_error(y_test[target_col], y_pred)
        mae = mean_absolute_error(y_test[target_col], y_pred)
        rmse = np.sqrt(mse)
        mape = np.mean(np.abs((y_test[target_col] - y_pred) / y_tes
        r2 = r2_score(y_test[target_col], y_pred)
        rse = np.sqrt(np.sum((y_test[target_col] - y_pred) ** 2) /

        print(f"{name} - MSE: {mse:.4f}, MAE: {mae:.4f}, RMSE: {rms

        results.append({
            'model': name,
            'target_col': target_col,
            'mse': mse,
            'mae': mae,
            'rmse': rmse,
```

```
            'mape': mape,
            'r2': r2,
            'rse': rse,
            'time': elapsed_time
        })

# Convert to DataFrame
results_df = pd.DataFrame(results)

# Plotting results
plt.figure(figsize=(14, 10))

# MSE
plt.subplot(2, 2, 1)
for name in results_df['model'].unique():
    model_results = results_df[results_df['model'] == name]
    plt.plot(model_results['target_col'], model_results['mse'], lab
plt.title('Mean Squared Error')
plt.xlabel('Target Variable')
plt.ylabel('MSE')
plt.xticks(rotation=45)
plt.legend()

# R^2
plt.subplot(2, 2, 2)
for name in results_df['model'].unique():
    model_results = results_df[results_df['model'] == name]
    plt.plot(model_results['target_col'], model_results['r2'], labe
plt.title('R^2 Score')
plt.xlabel('Target Variable')
plt.ylabel('R^2')
plt.xticks(rotation=45)
plt.legend()

# Time
plt.subplot(2, 2, 3)
for name in results_df['model'].unique():
    model_results = results_df[results_df['model'] == name]
    if 'time' in model_results.columns:
        plt.plot(model_results['target_col'], model_results['time']
plt.title('Time Taken')
plt.xlabel('Target Variable')
plt.ylabel('Time (s)')
plt.xticks(rotation=45)
plt.legend()

# RMSE
plt.subplot(2, 2, 4)
for name in results_df['model'].unique():
    model_results = results_df[results_df['model'] == name]
    plt.plot(model_results['target_col'], model_results['rmse'], la
plt.title('Root Mean Squared Error')
plt.xlabel('Target Variable')
plt.ylabel('RMSE')
plt.xticks(rotation=45)
plt.legend()
```

```
plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/numpy/_core/fromnumeric.py:5
7: FutureWarning: 'DataFrame.swapaxes' is deprecated and will be rem
oved in a future version. Please use 'DataFrame.transpose' instead.
  return bound(*args, **kwds)
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.011217 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560634
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.017887 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582565
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhe
ad of testing was 0.033130 seconds.
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652507
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010569 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680321
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010523 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637388
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010375 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.664167
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
```

```
ad of testing was 0.012354 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347749
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010640 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431678
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010609 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.269855
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.015694 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302631
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008610 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560517
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.015057 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582592
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.009560 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.653578
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
```

```
ad of testing was 0.008766 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680966
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008750 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.638017
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008430 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663916
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008622 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347473
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008694 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.432815
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.013348 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.271818
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008766 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.303505
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
```

```
ad of testing was 0.011562 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.559775
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010471 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.581604
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010682 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652370
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhe
ad of testing was 0.019319 seconds.
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.679292
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010532 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637137
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.015479 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663156
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010523 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.346177
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010567 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.430175
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010562 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.269549
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.010490 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 1101
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302539
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008772 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560990
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008705 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.583033
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.014160 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652748
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008734 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680719
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008882 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637060
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhe
ad of testing was 0.019909 seconds.
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663635
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.009162 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347197
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008703 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431502
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.008807 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.271324
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.013187 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 1099
[LightGBM] [Info] Number of data points in the train set: 107573, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.303219
Results for Linear Regression:
 ST58TA - Averaged MSE: 0.0287, MAE: 0.1311, RMSE: 0.1695, MAPE: 5.2
7%, R2: 0.2937, RSE: 0.1695
 ST117TA - Averaged MSE: 0.0287, MAE: 0.1303, RMSE: 0.1695, MAPE: 5.
15%, R2: 0.3011, RSE: 0.1695
 ST40TA - Averaged MSE: 0.0287, MAE: 0.1273, RMSE: 0.1693, MAPE: 4.9
5%, R2: 0.1904, RSE: 0.1693
 ST99TA - Averaged MSE: 0.0277, MAE: 0.1256, RMSE: 0.1664, MAPE: 4.8
1%, R2: 0.1649, RSE: 0.1664
 ST32TA - Averaged MSE: 0.0302, MAE: 0.1328, RMSE: 0.1737, MAPE: 5.1
6%, R2: 0.1293, RSE: 0.1737
```

```
 ST91TA - Averaged MSE: 0.0282, MAE: 0.1288, RMSE: 0.1680, MAPE: 4.9
8%, R2: 0.1500, RSE: 0.1680
 ST60TA - Averaged MSE: 0.1342, MAE: 0.2705, RMSE: 0.3664, MAPE: 8.8
2%, R2: 0.0553, RSE: 0.3664
 ST119TA - Averaged MSE: 0.1285, MAE: 0.2619, RMSE: 0.3585, MAPE: 8.
41%, R2: 0.1191, RSE: 0.3585
 ST62TA - Averaged MSE: 0.0389, MAE: 0.1573, RMSE: 0.1974, MAPE: 7.1
0%, R2: 0.1289, RSE: 0.1974
 ST121TA - Averaged MSE: 0.0398, MAE: 0.1570, RMSE: 0.1995, MAPE: 6.
94%, R2: 0.1548, RSE: 0.1995
Results for Decision Tree:
 ST58TA - Averaged MSE: 0.0042, MAE: 0.0484, RMSE: 0.0651, MAPE: 1.9
1%, R2: 0.8960, RSE: 0.0651
 ST117TA - Averaged MSE: 0.0038, MAE: 0.0444, RMSE: 0.0613, MAPE: 1.
74%, R2: 0.9087, RSE: 0.0613
 ST40TA - Averaged MSE: 0.0043, MAE: 0.0468, RMSE: 0.0654, MAPE: 1.7
8%, R2: 0.8792, RSE: 0.0654
 ST99TA - Averaged MSE: 0.0039, MAE: 0.0447, RMSE: 0.0625, MAPE: 1.7
0%, R2: 0.8820, RSE: 0.0625
 ST32TA - Averaged MSE: 0.0053, MAE: 0.0527, RMSE: 0.0728, MAPE: 2.0
1%, R2: 0.8473, RSE: 0.0728
 ST91TA - Averaged MSE: 0.0059, MAE: 0.0549, RMSE: 0.0769, MAPE: 2.1
1%, R2: 0.8217, RSE: 0.0769
 ST60TA - Averaged MSE: 0.0276, MAE: 0.1221, RMSE: 0.1663, MAPE: 3.8
3%, R2: 0.8054, RSE: 0.1663
 ST119TA - Averaged MSE: 0.0324, MAE: 0.1314, RMSE: 0.1801, MAPE: 3.
96%, R2: 0.7778, RSE: 0.1801
 ST62TA - Averaged MSE: 0.0067, MAE: 0.0615, RMSE: 0.0820, MAPE: 2.7
4%, R2: 0.8497, RSE: 0.0820
 ST121TA - Averaged MSE: 0.0081, MAE: 0.0664, RMSE: 0.0898, MAPE: 2.
92%, R2: 0.8287, RSE: 0.0898
Results for Random Forest:
 ST58TA - Averaged MSE: 0.0042, MAE: 0.0482, RMSE: 0.0648, MAPE: 1.9
1%, R2: 0.8969, RSE: 0.0648
 ST117TA - Averaged MSE: 0.0037, MAE: 0.0442, RMSE: 0.0610, MAPE: 1.
73%, R2: 0.9096, RSE: 0.0610
 ST40TA - Averaged MSE: 0.0042, MAE: 0.0466, RMSE: 0.0650, MAPE: 1.7
8%, R2: 0.8805, RSE: 0.0650
 ST99TA - Averaged MSE: 0.0039, MAE: 0.0446, RMSE: 0.0622, MAPE: 1.6
9%, R2: 0.8831, RSE: 0.0622
 ST32TA - Averaged MSE: 0.0052, MAE: 0.0525, RMSE: 0.0724, MAPE: 2.0
0%, R2: 0.8490, RSE: 0.0724
 ST91TA - Averaged MSE: 0.0059, MAE: 0.0547, RMSE: 0.0765, MAPE: 2.1
0%, R2: 0.8235, RSE: 0.0765
 ST60TA - Averaged MSE: 0.0273, MAE: 0.1216, RMSE: 0.1653, MAPE: 3.8
1%, R2: 0.8078, RSE: 0.1653
 ST119TA - Averaged MSE: 0.0321, MAE: 0.1310, RMSE: 0.1792, MAPE: 3.
95%, R2: 0.7800, RSE: 0.1792
 ST62TA - Averaged MSE: 0.0067, MAE: 0.0612, RMSE: 0.0816, MAPE: 2.7
3%, R2: 0.8511, RSE: 0.0816
 ST121TA - Averaged MSE: 0.0080, MAE: 0.0661, RMSE: 0.0894, MAPE: 2.
91%, R2: 0.8304, RSE: 0.0894
Results for k-NN:
 ST58TA - Averaged MSE: 0.0045, MAE: 0.0494, RMSE: 0.0674, MAPE: 1.9
6%, R2: 0.8884, RSE: 0.0674
 ST117TA - Averaged MSE: 0.0040, MAE: 0.0453, RMSE: 0.0634, MAPE: 1.
```

```
78%, R2: 0.9021, RSE: 0.0634
 ST40TA - Averaged MSE: 0.0045, MAE: 0.0475, RMSE: 0.0672, MAPE: 1.8
1%, R2: 0.8724, RSE: 0.0672
 ST99TA - Averaged MSE: 0.0041, MAE: 0.0455, RMSE: 0.0642, MAPE: 1.7
3%, R2: 0.8758, RSE: 0.0642
 ST32TA - Averaged MSE: 0.0055, MAE: 0.0533, RMSE: 0.0742, MAPE: 2.0
3%, R2: 0.8413, RSE: 0.0742
 ST91TA - Averaged MSE: 0.0061, MAE: 0.0555, RMSE: 0.0781, MAPE: 2.1
3%, R2: 0.8162, RSE: 0.0781
 ST60TA - Averaged MSE: 0.0284, MAE: 0.1233, RMSE: 0.1687, MAPE: 3.8
7%, R2: 0.7998, RSE: 0.1687
 ST119TA - Averaged MSE: 0.0335, MAE: 0.1326, RMSE: 0.1829, MAPE: 4.
02%, R2: 0.7707, RSE: 0.1829
 ST62TA - Averaged MSE: 0.0070, MAE: 0.0624, RMSE: 0.0839, MAPE: 2.7
9%, R2: 0.8427, RSE: 0.0839
 ST121TA - Averaged MSE: 0.0084, MAE: 0.0673, RMSE: 0.0917, MAPE: 2.
97%, R2: 0.8216, RSE: 0.0917
Results for Gradient Boosting:
 ST58TA - Averaged MSE: 0.0141, MAE: 0.0923, RMSE: 0.1188, MAPE: 3.7
0%, R2: 0.6533, RSE: 0.1188
 ST117TA - Averaged MSE: 0.0128, MAE: 0.0869, RMSE: 0.1132, MAPE: 3.
45%, R2: 0.6879, RSE: 0.1132
 ST40TA - Averaged MSE: 0.0143, MAE: 0.0898, RMSE: 0.1197, MAPE: 3.4
8%, R2: 0.5954, RSE: 0.1197
 ST99TA - Averaged MSE: 0.0131, MAE: 0.0855, RMSE: 0.1144, MAPE: 3.2
8%, R2: 0.6053, RSE: 0.1144
 ST32TA - Averaged MSE: 0.0161, MAE: 0.0958, RMSE: 0.1269, MAPE: 3.7
0%, R2: 0.5355, RSE: 0.1269
 ST91TA - Averaged MSE: 0.0153, MAE: 0.0914, RMSE: 0.1236, MAPE: 3.5
3%, R2: 0.5394, RSE: 0.1236
 ST60TA - Averaged MSE: 0.0809, MAE: 0.2106, RMSE: 0.2845, MAPE: 6.7
9%, R2: 0.4304, RSE: 0.2845
 ST119TA - Averaged MSE: 0.0848, MAE: 0.2146, RMSE: 0.2911, MAPE: 6.
81%, R2: 0.4190, RSE: 0.2911
 ST62TA - Averaged MSE: 0.0219, MAE: 0.1151, RMSE: 0.1478, MAPE: 5.1
8%, R2: 0.5111, RSE: 0.1478
 ST121TA - Averaged MSE: 0.0239, MAE: 0.1170, RMSE: 0.1544, MAPE: 5.
18%, R2: 0.4934, RSE: 0.1544
Results for XGBoost:
 ST58TA - Averaged MSE: 0.0043, MAE: 0.0493, RMSE: 0.0656, MAPE: 1.9
5%, R2: 0.8942, RSE: 0.0656
 ST117TA - Averaged MSE: 0.0039, MAE: 0.0456, RMSE: 0.0621, MAPE: 1.
79%, R2: 0.9061, RSE: 0.0621
 ST40TA - Averaged MSE: 0.0043, MAE: 0.0478, RMSE: 0.0659, MAPE: 1.8
2%, R2: 0.8773, RSE: 0.0659
 ST99TA - Averaged MSE: 0.0040, MAE: 0.0461, RMSE: 0.0634, MAPE: 1.7
5%, R2: 0.8787, RSE: 0.0634
 ST32TA - Averaged MSE: 0.0054, MAE: 0.0537, RMSE: 0.0732, MAPE: 2.0
5%, R2: 0.8457, RSE: 0.0732
 ST91TA - Averaged MSE: 0.0059, MAE: 0.0556, RMSE: 0.0768, MAPE: 2.1
3%, R2: 0.8224, RSE: 0.0768
 ST60TA - Averaged MSE: 0.0278, MAE: 0.1239, RMSE: 0.1668, MAPE: 3.8
9%, R2: 0.8041, RSE: 0.1668
 ST119TA - Averaged MSE: 0.0326, MAE: 0.1334, RMSE: 0.1805, MAPE: 4.
04%, R2: 0.7767, RSE: 0.1805
 ST62TA - Averaged MSE: 0.0069, MAE: 0.0627, RMSE: 0.0828, MAPE: 2.8
```

```
0%, R2: 0.8466, RSE: 0.0828
 ST121TA - Averaged MSE: 0.0083, MAE: 0.0679, RMSE: 0.0910, MAPE: 2.
99%, R2: 0.8240, RSE: 0.0910
Results for CatBoost:
 ST58TA - Averaged MSE: 0.0045, MAE: 0.0506, RMSE: 0.0673, MAPE: 2.0
1%, R2: 0.8886, RSE: 0.0673
 ST117TA - Averaged MSE: 0.0040, MAE: 0.0466, RMSE: 0.0632, MAPE: 1.
83%, R2: 0.9027, RSE: 0.0632
 ST40TA - Averaged MSE: 0.0045, MAE: 0.0486, RMSE: 0.0669, MAPE: 1.8
6%, R2: 0.8736, RSE: 0.0669
 ST99TA - Averaged MSE: 0.0041, MAE: 0.0468, RMSE: 0.0642, MAPE: 1.7
8%, R2: 0.8757, RSE: 0.0642
 ST32TA - Averaged MSE: 0.0055, MAE: 0.0547, RMSE: 0.0741, MAPE: 2.0
9%, R2: 0.8415, RSE: 0.0741
 ST91TA - Averaged MSE: 0.0060, MAE: 0.0567, RMSE: 0.0777, MAPE: 2.1
8%, R2: 0.8179, RSE: 0.0777
 ST60TA - Averaged MSE: 0.0284, MAE: 0.1253, RMSE: 0.1684, MAPE: 3.9
3%, R2: 0.8004, RSE: 0.1684
 ST119TA - Averaged MSE: 0.0332, MAE: 0.1351, RMSE: 0.1821, MAPE: 4.
10%, R2: 0.7728, RSE: 0.1821
 ST62TA - Averaged MSE: 0.0071, MAE: 0.0640, RMSE: 0.0844, MAPE: 2.8
6%, R2: 0.8408, RSE: 0.0844
 ST121TA - Averaged MSE: 0.0086, MAE: 0.0691, RMSE: 0.0926, MAPE: 3.
05%, R2: 0.8180, RSE: 0.0926
Results for LightGBM:
 ST58TA - Averaged MSE: 0.0060, MAE: 0.0586, RMSE: 0.0777, MAPE: 2.3
3%, R2: 0.8517, RSE: 0.0777
 ST117TA - Averaged MSE: 0.0055, MAE: 0.0546, RMSE: 0.0739, MAPE: 2.
15%, R2: 0.8671, RSE: 0.0739
 ST40TA - Averaged MSE: 0.0060, MAE: 0.0565, RMSE: 0.0772, MAPE: 2.1
6%, R2: 0.8318, RSE: 0.0772
 ST99TA - Averaged MSE: 0.0056, MAE: 0.0546, RMSE: 0.0745, MAPE: 2.0
8%, R2: 0.8323, RSE: 0.0745
 ST32TA - Averaged MSE: 0.0071, MAE: 0.0625, RMSE: 0.0841, MAPE: 2.3
9%, R2: 0.7958, RSE: 0.0841
 ST91TA - Averaged MSE: 0.0077, MAE: 0.0645, RMSE: 0.0877, MAPE: 2.4
8%, R2: 0.7684, RSE: 0.0877
 ST60TA - Averaged MSE: 0.0354, MAE: 0.1405, RMSE: 0.1881, MAPE: 4.4
2%, R2: 0.7511, RSE: 0.1881
 ST119TA - Averaged MSE: 0.0405, MAE: 0.1502, RMSE: 0.2013, MAPE: 4.
60%, R2: 0.7222, RSE: 0.2013
 ST62TA - Averaged MSE: 0.0097, MAE: 0.0756, RMSE: 0.0987, MAPE: 3.3
8%, R2: 0.7820, RSE: 0.0987
 ST121TA - Averaged MSE: 0.0117, MAE: 0.0808, RMSE: 0.1082, MAPE: 3.
58%, R2: 0.7512, RSE: 0.1082

Results for target variable: ST58TA
Linear Regression - MSE: 0.0287, MAE: 0.1311, RMSE: 0.1695, MAPE: 5.
27%, R2: 0.2937, RSE: 0.1695, Time: 0.22s
Decision Tree - MSE: 0.0042, MAE: 0.0484, RMSE: 0.0651, MAPE: 1.92%,
R2: 0.8959, RSE: 0.0651, Time: 1.67s
Random Forest - MSE: 0.0042, MAE: 0.0484, RMSE: 0.0650, MAPE: 1.91%,
R2: 0.8960, RSE: 0.0650, Time: 127.58s
k-NN - MSE: 0.0045, MAE: 0.0494, RMSE: 0.0672, MAPE: 1.95%, R2: 0.88
91, RSE: 0.0672, Time: 7.54s
Gradient Boosting - MSE: 0.0138, MAE: 0.0913, RMSE: 0.1174, MAPE: 3.
```

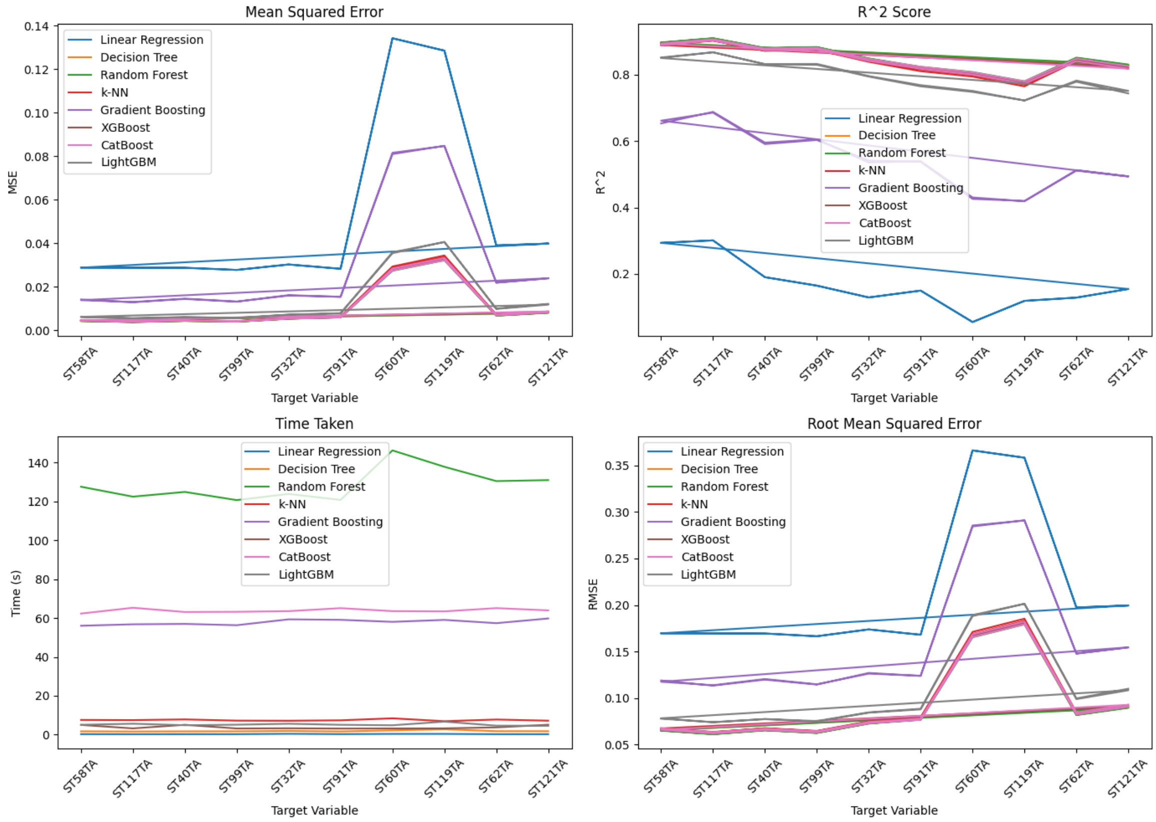
```
66%, R2: 0.6614, RSE: 0.1174, Time: 56.03s
XGBoost - MSE: 0.0043, MAE: 0.0495, RMSE: 0.0659, MAPE: 1.96%, R2:
0.8933, RSE: 0.0659, Time: 4.97s
CatBoost - MSE: 0.0043, MAE: 0.0494, RMSE: 0.0657, MAPE: 1.96%, R2:
0.8939, RSE: 0.0657, Time: 62.29s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.061071 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
LightGBM - MSE: 0.0061, MAE: 0.0588, RMSE: 0.0781, MAPE: 2.33%, R2:
0.8501, RSE: 0.0781, Time: 5.03s

Results for target variable: ST117TA
Linear Regression - MSE: 0.0287, MAE: 0.1303, RMSE: 0.1695, MAPE: 5.
15%, R2: 0.3011, RSE: 0.1695, Time: 0.19s
Decision Tree - MSE: 0.0037, MAE: 0.0444, RMSE: 0.0612, MAPE: 1.74%,
R2: 0.9090, RSE: 0.0612, Time: 1.58s
Random Forest - MSE: 0.0037, MAE: 0.0443, RMSE: 0.0611, MAPE: 1.74%,
R2: 0.9090, RSE: 0.0611, Time: 122.47s
k-NN - MSE: 0.0040, MAE: 0.0455, RMSE: 0.0631, MAPE: 1.78%, R2: 0.90
31, RSE: 0.0631, Time: 7.45s
Gradient Boosting - MSE: 0.0129, MAE: 0.0872, RMSE: 0.1137, MAPE: 3.
47%, R2: 0.6856, RSE: 0.1137, Time: 56.76s
XGBoost - MSE: 0.0039, MAE: 0.0457, RMSE: 0.0622, MAPE: 1.80%, R2:
0.9057, RSE: 0.0622, Time: 3.19s
CatBoost - MSE: 0.0038, MAE: 0.0457, RMSE: 0.0619, MAPE: 1.79%, R2:
0.9067, RSE: 0.0619, Time: 65.27s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.061330 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
LightGBM - MSE: 0.0054, MAE: 0.0547, RMSE: 0.0738, MAPE: 2.16%, R2:
0.8674, RSE: 0.0738, Time: 5.59s

Results for target variable: ST40TA
Linear Regression - MSE: 0.0287, MAE: 0.1273, RMSE: 0.1693, MAPE: 4.
95%, R2: 0.1904, RSE: 0.1693, Time: 0.20s
Decision Tree - MSE: 0.0043, MAE: 0.0468, RMSE: 0.0654, MAPE: 1.78%,
R2: 0.8793, RSE: 0.0654, Time: 1.65s
Random Forest - MSE: 0.0043, MAE: 0.0468, RMSE: 0.0653, MAPE: 1.78%,
R2: 0.8794, RSE: 0.0653, Time: 124.90s
k-NN - MSE: 0.0045, MAE: 0.0479, RMSE: 0.0674, MAPE: 1.83%, R2: 0.87
16, RSE: 0.0674, Time: 7.80s
Gradient Boosting - MSE: 0.0145, MAE: 0.0901, RMSE: 0.1203, MAPE: 3.
49%, R2: 0.5910, RSE: 0.1203, Time: 56.96s
XGBoost - MSE: 0.0044, MAE: 0.0480, RMSE: 0.0662, MAPE: 1.83%, R2:
0.8761, RSE: 0.0662, Time: 5.03s
CatBoost - MSE: 0.0043, MAE: 0.0476, RMSE: 0.0655, MAPE: 1.82%, R2:
```

```
0.8788, RSE: 0.0655, Time: 63.10s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.049409 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
LightGBM - MSE: 0.0060, MAE: 0.0567, RMSE: 0.0773, MAPE: 2.17%, R2:
0.8313, RSE: 0.0773, Time: 4.82s

Results for target variable: ST99TA
Linear Regression - MSE: 0.0277, MAE: 0.1256, RMSE: 0.1664, MAPE: 4.
81%, R2: 0.1649, RSE: 0.1664, Time: 0.21s
Decision Tree - MSE: 0.0039, MAE: 0.0447, RMSE: 0.0625, MAPE: 1.70%,
R2: 0.8823, RSE: 0.0625, Time: 1.69s
Random Forest - MSE: 0.0039, MAE: 0.0447, RMSE: 0.0624, MAPE: 1.70%,
R2: 0.8824, RSE: 0.0624, Time: 120.69s
k-NN - MSE: 0.0042, MAE: 0.0457, RMSE: 0.0645, MAPE: 1.74%, R2: 0.87
44, RSE: 0.0645, Time: 7.17s
Gradient Boosting - MSE: 0.0131, MAE: 0.0857, RMSE: 0.1147, MAPE: 3.
29%, R2: 0.6033, RSE: 0.1147, Time: 56.30s
XGBoost - MSE: 0.0041, MAE: 0.0465, RMSE: 0.0637, MAPE: 1.77%, R2:
0.8776, RSE: 0.0637, Time: 3.14s
CatBoost - MSE: 0.0040, MAE: 0.0458, RMSE: 0.0629, MAPE: 1.74%, R2:
0.8808, RSE: 0.0629, Time: 63.20s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.044250 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
LightGBM - MSE: 0.0056, MAE: 0.0552, RMSE: 0.0751, MAPE: 2.10%, R2:
0.8296, RSE: 0.0751, Time: 5.09s

Results for target variable: ST32TA
Linear Regression - MSE: 0.0302, MAE: 0.1328, RMSE: 0.1737, MAPE: 5.
16%, R2: 0.1293, RSE: 0.1737, Time: 0.40s
Decision Tree - MSE: 0.0053, MAE: 0.0526, RMSE: 0.0727, MAPE: 2.01%,
R2: 0.8476, RSE: 0.0727, Time: 1.91s
Random Forest - MSE: 0.0053, MAE: 0.0526, RMSE: 0.0726, MAPE: 2.01%,
R2: 0.8478, RSE: 0.0726, Time: 123.87s
k-NN - MSE: 0.0056, MAE: 0.0536, RMSE: 0.0748, MAPE: 2.04%, R2: 0.83
86, RSE: 0.0748, Time: 7.11s
Gradient Boosting - MSE: 0.0159, MAE: 0.0950, RMSE: 0.1261, MAPE: 3.
67%, R2: 0.5414, RSE: 0.1261, Time: 59.32s
XGBoost - MSE: 0.0054, MAE: 0.0539, RMSE: 0.0733, MAPE: 2.06%, R2:
0.8450, RSE: 0.0733, Time: 3.24s
CatBoost - MSE: 0.0053, MAE: 0.0535, RMSE: 0.0728, MAPE: 2.04%, R2:
0.8473, RSE: 0.0728, Time: 63.53s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.042766 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
LightGBM - MSE: 0.0071, MAE: 0.0630, RMSE: 0.0845, MAPE: 2.41%, R2:
0.7942, RSE: 0.0845, Time: 5.58s

Results for target variable: ST91TA
Linear Regression - MSE: 0.0282, MAE: 0.1288, RMSE: 0.1680, MAPE: 4.
98%, R2: 0.1500, RSE: 0.1680, Time: 0.22s
Decision Tree - MSE: 0.0059, MAE: 0.0549, RMSE: 0.0768, MAPE: 2.11%,
R2: 0.8221, RSE: 0.0768, Time: 1.61s
Random Forest - MSE: 0.0059, MAE: 0.0548, RMSE: 0.0768, MAPE: 2.10%,
R2: 0.8222, RSE: 0.0768, Time: 120.79s
k-NN - MSE: 0.0063, MAE: 0.0562, RMSE: 0.0793, MAPE: 2.15%, R2: 0.81
04, RSE: 0.0793, Time: 7.34s
Gradient Boosting - MSE: 0.0153, MAE: 0.0916, RMSE: 0.1239, MAPE: 3.
53%, R2: 0.5377, RSE: 0.1239, Time: 59.07s
XGBoost - MSE: 0.0060, MAE: 0.0561, RMSE: 0.0772, MAPE: 2.15%, R2:
0.8204, RSE: 0.0772, Time: 3.10s
CatBoost - MSE: 0.0058, MAE: 0.0556, RMSE: 0.0765, MAPE: 2.13%, R2:
0.8238, RSE: 0.0765, Time: 65.07s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.042317 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
LightGBM - MSE: 0.0078, MAE: 0.0651, RMSE: 0.0884, MAPE: 2.50%, R2:
0.7648, RSE: 0.0884, Time: 5.03s

Results for target variable: ST60TA
Linear Regression - MSE: 0.1342, MAE: 0.2705, RMSE: 0.3664, MAPE: 8.
82%, R2: 0.0553, RSE: 0.3664, Time: 0.34s
Decision Tree - MSE: 0.0277, MAE: 0.1221, RMSE: 0.1663, MAPE: 3.83%,
R2: 0.8054, RSE: 0.1663, Time: 2.18s
Random Forest - MSE: 0.0276, MAE: 0.1220, RMSE: 0.1661, MAPE: 3.82%,
R2: 0.8057, RSE: 0.1661, Time: 146.28s
k-NN - MSE: 0.0293, MAE: 0.1250, RMSE: 0.1710, MAPE: 3.91%, R2: 0.79
41, RSE: 0.1710, Time: 8.37s
Gradient Boosting - MSE: 0.0816, MAE: 0.2111, RMSE: 0.2856, MAPE: 6.
82%, R2: 0.4259, RSE: 0.2856, Time: 58.03s
XGBoost - MSE: 0.0282, MAE: 0.1250, RMSE: 0.1678, MAPE: 3.92%, R2:
0.8018, RSE: 0.1678, Time: 3.15s
CatBoost - MSE: 0.0274, MAE: 0.1233, RMSE: 0.1656, MAPE: 3.87%, R2:
0.8069, RSE: 0.1656, Time: 63.54s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.042176 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
```

```
[LightGBM] [Info] Start training from score 3.347149
LightGBM - MSE: 0.0358, MAE: 0.1413, RMSE: 0.1892, MAPE: 4.45%, R2: 
0.7481, RSE: 0.1892, Time: 4.79s

Results for target variable: ST119TA
Linear Regression - MSE: 0.1285, MAE: 0.2619, RMSE: 0.3585, MAPE: 8.
41%, R2: 0.1191, RSE: 0.3585, Time: 0.34s
Decision Tree - MSE: 0.0325, MAE: 0.1315, RMSE: 0.1803, MAPE: 3.97%, 
R2: 0.7773, RSE: 0.1803, Time: 2.75s
Random Forest - MSE: 0.0324, MAE: 0.1314, RMSE: 0.1801, MAPE: 3.96%, 
R2: 0.7776, RSE: 0.1801, Time: 137.84s
k-NN - MSE: 0.0343, MAE: 0.1341, RMSE: 0.1853, MAPE: 4.04%, R2: 0.76
48, RSE: 0.1853, Time: 6.88s
Gradient Boosting - MSE: 0.0846, MAE: 0.2145, RMSE: 0.2909, MAPE: 6.
80%, R2: 0.4198, RSE: 0.2909, Time: 59.03s
XGBoost - MSE: 0.0326, MAE: 0.1335, RMSE: 0.1804, MAPE: 4.04%, R2: 
0.7769, RSE: 0.1804, Time: 3.19s
CatBoost - MSE: 0.0322, MAE: 0.1327, RMSE: 0.1793, MAPE: 4.02%, R2: 
0.7795, RSE: 0.1793, Time: 63.42s
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhe
ad of testing was 0.077088 seconds.
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
LightGBM - MSE: 0.0405, MAE: 0.1503, RMSE: 0.2013, MAPE: 4.60%, R2: 
0.7222, RSE: 0.2013, Time: 6.67s

Results for target variable: ST62TA
Linear Regression - MSE: 0.0389, MAE: 0.1573, RMSE: 0.1974, MAPE: 7.
10%, R2: 0.1289, RSE: 0.1974, Time: 0.20s
Decision Tree - MSE: 0.0067, MAE: 0.0615, RMSE: 0.0820, MAPE: 2.74%, 
R2: 0.8496, RSE: 0.0820, Time: 1.75s
Random Forest - MSE: 0.0067, MAE: 0.0615, RMSE: 0.0819, MAPE: 2.74%, 
R2: 0.8499, RSE: 0.0819, Time: 130.48s
k-NN - MSE: 0.0071, MAE: 0.0629, RMSE: 0.0844, MAPE: 2.81%, R2: 0.84
06, RSE: 0.0844, Time: 7.74s
Gradient Boosting - MSE: 0.0218, MAE: 0.1147, RMSE: 0.1476, MAPE: 5.
17%, R2: 0.5126, RSE: 0.1476, Time: 57.35s
XGBoost - MSE: 0.0069, MAE: 0.0630, RMSE: 0.0831, MAPE: 2.81%, R2: 
0.8456, RSE: 0.0831, Time: 3.70s
CatBoost - MSE: 0.0068, MAE: 0.0624, RMSE: 0.0824, MAPE: 2.79%, R2: 
0.8483, RSE: 0.0824, Time: 65.09s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.042551 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
LightGBM - MSE: 0.0099, MAE: 0.0762, RMSE: 0.0995, MAPE: 3.41%, R2: 
0.7786, RSE: 0.0995, Time: 4.58s

Results for target variable: ST121TA
```

```
Linear Regression - MSE: 0.0398, MAE: 0.1570, RMSE: 0.1995, MAPE: 6.
94%, R2: 0.1548, RSE: 0.1995, Time: 0.19s
Decision Tree - MSE: 0.0081, MAE: 0.0665, RMSE: 0.0899, MAPE: 2.93%,
R2: 0.8283, RSE: 0.0899, Time: 1.73s
Random Forest - MSE: 0.0081, MAE: 0.0664, RMSE: 0.0898, MAPE: 2.92%,
R2: 0.8286, RSE: 0.0898, Time: 130.98s
k-NN - MSE: 0.0086, MAE: 0.0681, RMSE: 0.0926, MAPE: 3.00%, R2: 0.81
79, RSE: 0.0926, Time: 7.15s
Gradient Boosting - MSE: 0.0238, MAE: 0.1168, RMSE: 0.1543, MAPE: 5.
17%, R2: 0.4941, RSE: 0.1543, Time: 59.76s
XGBoost - MSE: 0.0082, MAE: 0.0677, RMSE: 0.0906, MAPE: 2.98%, R2:
0.8256, RSE: 0.0906, Time: 5.07s
CatBoost - MSE: 0.0082, MAE: 0.0677, RMSE: 0.0906, MAPE: 2.98%, R2:
0.8259, RSE: 0.0906, Time: 63.94s
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhe
ad of testing was 0.043421 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 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
LightGBM - MSE: 0.0121, MAE: 0.0824, RMSE: 0.1099, MAPE: 3.64%, R2:
0.7435, RSE: 0.1099, Time: 4.52s
```



In [ ]:
```python
import pandas as pd
import numpy as np
import time
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostin
from sklearn.neighbors import KNeighborsRegressor
```

```
from sklearn.metrics import mean_squared_error, r2_score, mean_abso
import xgboost as xgb
from catboost import CatBoostRegressor
import lightgbm as lgb
import matplotlib.pyplot as plt

# Assuming 'X' and 'y' are defined before this point
# Example:
# X = df.drop(columns=target_columns)
# y = df[target_columns]  # where target_columns is a list of targe

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size

# Initialize base models
base_models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'k-NN': KNeighborsRegressor(n_neighbors=10),
    'Gradient Boosting': GradientBoostingRegressor(),
    'XGBoost': xgb.XGBRegressor(objective='reg:squarederror'),
    'CatBoost': CatBoostRegressor(verbose=0),
    'LightGBM': lgb.LGBMRegressor(force_col_wise=True)
}

results = []

# Train and evaluate each base model
for target_col in y_train.columns:
    print(f"Results for target variable: {target_col}")
    for name, model in base_models.items():
        start_time = time.time()
        model.fit(X_train, y_train[target_col])
        y_pred = model.predict(X_test)
        end_time = time.time()

        mse = mean_squared_error(y_test[target_col], y_pred)
        mae = mean_absolute_error(y_test[target_col], y_pred)
        rmse = np.sqrt(mse)
        mape = np.mean(np.abs((y_test[target_col] - y_pred) / y_tes
        r2 = r2_score(y_test[target_col], y_pred)
        rse = np.sqrt(np.sum((y_test[target_col] - y_pred) ** 2) /

        print(f"{name} - MSE: {mse:.4f}, MAE: {mae:.4f}, RMSE: {rms

        results.append({
            'model': name,
            'target_col': target_col,
            'mse': mse,
            'mae': mae,
            'rmse': rmse,
            'mape': mape,
            'r2': r2,
            'rse': rse,
            'time': end_time - start_time
```

```
        })

# Create Voting Regressor
voting_regressor = VotingRegressor(estimators=[
    ('lr', base_models['Linear Regression']),
    ('dt', base_models['Decision Tree']),
    ('rf', base_models['Random Forest']),
    ('knn', base_models['k-NN']),
    ('gb', base_models['Gradient Boosting']),
    ('xgb', base_models['XGBoost']),
    ('catboost', base_models['CatBoost']),
    ('lgbm', base_models['LightGBM'])
])

# Train and evaluate Voting Regressor
for target_col in y_train.columns:
    print(f"\nVoting Regressor Results for target variable: {target_
    start_time = time.time()
    voting_regressor.fit(X_train, y_train[target_col])
    y_pred = voting_regressor.predict(X_test)
    end_time = time.time()

    mse = mean_squared_error(y_test[target_col], y_pred)
    mae = mean_absolute_error(y_test[target_col], y_pred)
    rmse = np.sqrt(mse)
    mape = np.mean(np.abs((y_test[target_col] - y_pred) / y_test[ta
    r2 = r2_score(y_test[target_col], y_pred)
    rse = np.sqrt(np.sum((y_test[target_col] - y_pred) ** 2) / (len

    print(f"Voting Regressor - MSE: {mse:.4f}, MAE: {mae:.4f}, RMSE

    results.append({
        'model': 'Voting Regressor',
        'target_col': target_col,
        'mse': mse,
        'mae': mae,
        'rmse': rmse,
        'mape': mape,
        'r2': r2,
        'rse': rse,
        'time': end_time - start_time
    })

# Convert results to DataFrame
results_df = pd.DataFrame(results)

# Plotting results
for metric in ['mse', 'r2', 'time', 'rmse']:
    plt.figure(figsize=(12, 6))
    for name in results_df['model'].unique():
        model_results = results_df[results_df['model'] == name]
        plt.plot(model_results['target_col'], model_results[metric]
    plt.title(f'{metric.upper()} for each model across target varia
    plt.xlabel('Target Variable')
    plt.ylabel(metric.upper())
    plt.legend()
```

```
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```

```
Results for target variable: ST58TA
Linear Regression - MSE: 0.0287, MAE: 0.1311, RMSE: 0.1695, MAPE: 5.
27%, R2: 0.2937, Time: 0.23s
Decision Tree - MSE: 0.0042, MAE: 0.0484, RMSE: 0.0651, MAPE: 1.92%,
R2: 0.8959, Time: 1.65s
Random Forest - MSE: 0.0042, MAE: 0.0484, RMSE: 0.0650, MAPE: 1.91%,
R2: 0.8960, Time: 126.40s
k-NN - MSE: 0.0045, MAE: 0.0494, RMSE: 0.0672, MAPE: 1.95%, R2: 0.88
91, Time: 7.72s
Gradient Boosting - MSE: 0.0138, MAE: 0.0913, RMSE: 0.1174, MAPE: 3.
66%, R2: 0.6614, Time: 60.64s
XGBoost - MSE: 0.0043, MAE: 0.0495, RMSE: 0.0659, MAPE: 1.96%, R2:
0.8933, Time: 3.16s
CatBoost - MSE: 0.0043, MAE: 0.0494, RMSE: 0.0657, MAPE: 1.96%, R2:
0.8939, Time: 63.32s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
LightGBM - MSE: 0.0061, MAE: 0.0588, RMSE: 0.0781, MAPE: 2.33%, R2:
0.8501, Time: 5.47s
Results for target variable: ST117TA
Linear Regression - MSE: 0.0287, MAE: 0.1303, RMSE: 0.1695, MAPE: 5.
15%, R2: 0.3011, Time: 0.36s
Decision Tree - MSE: 0.0037, MAE: 0.0444, RMSE: 0.0612, MAPE: 1.74%,
R2: 0.9090, Time: 1.79s
Random Forest - MSE: 0.0037, MAE: 0.0443, RMSE: 0.0611, MAPE: 1.74%,
R2: 0.9091, Time: 131.52s
k-NN - MSE: 0.0040, MAE: 0.0455, RMSE: 0.0631, MAPE: 1.78%, R2: 0.90
31, Time: 7.94s
Gradient Boosting - MSE: 0.0129, MAE: 0.0872, RMSE: 0.1137, MAPE: 3.
47%, R2: 0.6856, Time: 57.02s
XGBoost - MSE: 0.0039, MAE: 0.0457, RMSE: 0.0622, MAPE: 1.80%, R2:
0.9057, Time: 4.91s
CatBoost - MSE: 0.0038, MAE: 0.0457, RMSE: 0.0619, MAPE: 1.79%, R2:
0.9067, Time: 65.06s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
LightGBM - MSE: 0.0054, MAE: 0.0547, RMSE: 0.0738, MAPE: 2.16%, R2:
0.8674, Time: 5.08s
Results for target variable: ST40TA
Linear Regression - MSE: 0.0287, MAE: 0.1273, RMSE: 0.1693, MAPE: 4.
95%, R2: 0.1904, Time: 0.22s
Decision Tree - MSE: 0.0043, MAE: 0.0468, RMSE: 0.0654, MAPE: 1.78%,
R2: 0.8793, Time: 1.71s
Random Forest - MSE: 0.0043, MAE: 0.0468, RMSE: 0.0654, MAPE: 1.78%,
R2: 0.8794, Time: 128.95s
k-NN - MSE: 0.0045, MAE: 0.0479, RMSE: 0.0674, MAPE: 1.83%, R2: 0.87
16, Time: 6.84s
Gradient Boosting - MSE: 0.0145, MAE: 0.0901, RMSE: 0.1203, MAPE: 3.
```

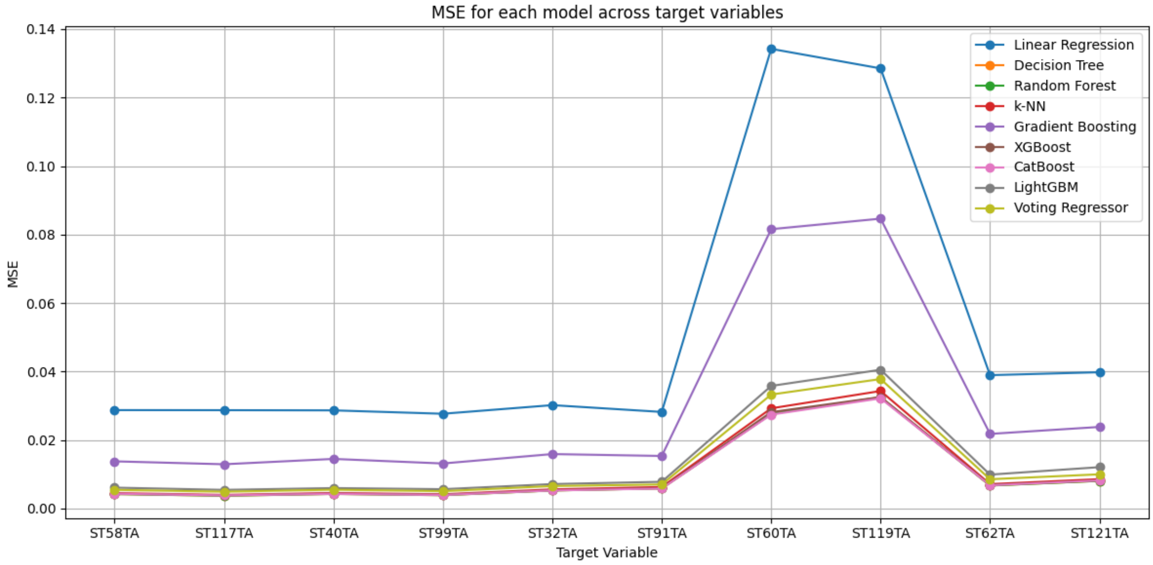
```
49%, R2: 0.5910, Time: 59.60s
XGBoost - MSE: 0.0044, MAE: 0.0480, RMSE: 0.0662, MAPE: 1.83%, R2:
0.8761, Time: 3.28s
CatBoost - MSE: 0.0043, MAE: 0.0476, RMSE: 0.0655, MAPE: 1.82%, R2:
0.8788, Time: 63.93s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
LightGBM - MSE: 0.0060, MAE: 0.0567, RMSE: 0.0773, MAPE: 2.17%, R2:
0.8313, Time: 5.78s
Results for target variable: ST99TA
Linear Regression - MSE: 0.0277, MAE: 0.1256, RMSE: 0.1664, MAPE: 4.
81%, R2: 0.1649, Time: 0.24s
Decision Tree - MSE: 0.0039, MAE: 0.0447, RMSE: 0.0625, MAPE: 1.70%,
R2: 0.8823, Time: 1.77s
Random Forest - MSE: 0.0039, MAE: 0.0447, RMSE: 0.0624, MAPE: 1.70%,
R2: 0.8824, Time: 130.89s
k-NN - MSE: 0.0042, MAE: 0.0457, RMSE: 0.0645, MAPE: 1.74%, R2: 0.87
44, Time: 7.89s
Gradient Boosting - MSE: 0.0131, MAE: 0.0857, RMSE: 0.1147, MAPE: 3.
29%, R2: 0.6033, Time: 60.36s
XGBoost - MSE: 0.0041, MAE: 0.0465, RMSE: 0.0637, MAPE: 1.77%, R2:
0.8776, Time: 3.20s
CatBoost - MSE: 0.0040, MAE: 0.0458, RMSE: 0.0629, MAPE: 1.74%, R2:
0.8808, Time: 64.09s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
LightGBM - MSE: 0.0056, MAE: 0.0552, RMSE: 0.0751, MAPE: 2.10%, R2:
0.8296, Time: 5.46s
Results for target variable: ST32TA
Linear Regression - MSE: 0.0302, MAE: 0.1328, RMSE: 0.1737, MAPE: 5.
16%, R2: 0.1293, Time: 0.33s
Decision Tree - MSE: 0.0053, MAE: 0.0526, RMSE: 0.0727, MAPE: 2.01%,
R2: 0.8476, Time: 2.02s
Random Forest - MSE: 0.0053, MAE: 0.0526, RMSE: 0.0726, MAPE: 2.01%,
R2: 0.8478, Time: 131.13s
k-NN - MSE: 0.0056, MAE: 0.0536, RMSE: 0.0748, MAPE: 2.04%, R2: 0.83
86, Time: 7.83s
Gradient Boosting - MSE: 0.0159, MAE: 0.0950, RMSE: 0.1261, MAPE: 3.
67%, R2: 0.5414, Time: 59.00s
XGBoost - MSE: 0.0054, MAE: 0.0539, RMSE: 0.0733, MAPE: 2.06%, R2:
0.8450, Time: 3.29s
CatBoost - MSE: 0.0053, MAE: 0.0535, RMSE: 0.0728, MAPE: 2.04%, R2:
0.8473, Time: 63.67s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
LightGBM - MSE: 0.0071, MAE: 0.0630, RMSE: 0.0845, MAPE: 2.41%, R2:
0.7942, Time: 5.07s
Results for target variable: ST91TA
Linear Regression - MSE: 0.0282, MAE: 0.1288, RMSE: 0.1680, MAPE: 4.
98%, R2: 0.1500, Time: 0.23s
```

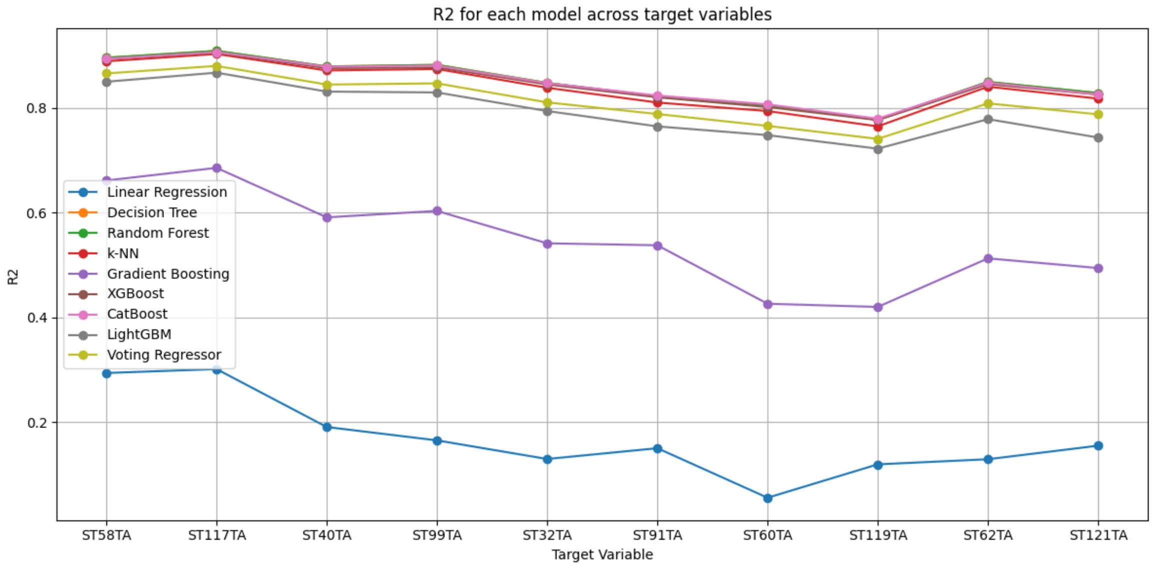
```
Decision Tree - MSE: 0.0059, MAE: 0.0549, RMSE: 0.0768, MAPE: 2.11%,
R2: 0.8221, Time: 2.12s
Random Forest - MSE: 0.0059, MAE: 0.0548, RMSE: 0.0768, MAPE: 2.10%,
R2: 0.8223, Time: 129.31s
k-NN - MSE: 0.0063, MAE: 0.0562, RMSE: 0.0793, MAPE: 2.15%, R2: 0.81
04, Time: 7.00s
Gradient Boosting - MSE: 0.0153, MAE: 0.0916, RMSE: 0.1239, MAPE: 3.
53%, R2: 0.5377, Time: 60.16s
XGBoost - MSE: 0.0060, MAE: 0.0561, RMSE: 0.0772, MAPE: 2.15%, R2:
0.8204, Time: 3.11s
CatBoost - MSE: 0.0058, MAE: 0.0556, RMSE: 0.0765, MAPE: 2.13%, R2:
0.8238, Time: 65.51s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
LightGBM - MSE: 0.0078, MAE: 0.0651, RMSE: 0.0884, MAPE: 2.50%, R2:
0.7648, Time: 5.55s
Results for target variable: ST60TA
Linear Regression - MSE: 0.1342, MAE: 0.2705, RMSE: 0.3664, MAPE: 8.
82%, R2: 0.0553, Time: 0.22s
Decision Tree - MSE: 0.0277, MAE: 0.1221, RMSE: 0.1663, MAPE: 3.83%,
R2: 0.8054, Time: 1.94s
Random Forest - MSE: 0.0276, MAE: 0.1220, RMSE: 0.1662, MAPE: 3.82%,
R2: 0.8056, Time: 148.71s
k-NN - MSE: 0.0293, MAE: 0.1250, RMSE: 0.1710, MAPE: 3.91%, R2: 0.79
41, Time: 7.87s
Gradient Boosting - MSE: 0.0816, MAE: 0.2111, RMSE: 0.2856, MAPE: 6.
82%, R2: 0.4259, Time: 62.15s
XGBoost - MSE: 0.0282, MAE: 0.1250, RMSE: 0.1678, MAPE: 3.92%, R2:
0.8018, Time: 3.29s
CatBoost - MSE: 0.0274, MAE: 0.1233, RMSE: 0.1656, MAPE: 3.87%, R2:
0.8069, Time: 63.37s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347149
LightGBM - MSE: 0.0358, MAE: 0.1413, RMSE: 0.1892, MAPE: 4.45%, R2:
0.7481, Time: 5.01s
Results for target variable: ST119TA
Linear Regression - MSE: 0.1285, MAE: 0.2619, RMSE: 0.3585, MAPE: 8.
41%, R2: 0.1191, Time: 0.22s
Decision Tree - MSE: 0.0325, MAE: 0.1315, RMSE: 0.1803, MAPE: 3.97%,
R2: 0.7773, Time: 1.91s
Random Forest - MSE: 0.0324, MAE: 0.1314, RMSE: 0.1801, MAPE: 3.96%,
R2: 0.7777, Time: 146.80s
k-NN - MSE: 0.0343, MAE: 0.1341, RMSE: 0.1853, MAPE: 4.04%, R2: 0.76
48, Time: 8.05s
Gradient Boosting - MSE: 0.0846, MAE: 0.2145, RMSE: 0.2909, MAPE: 6.
80%, R2: 0.4198, Time: 60.47s
XGBoost - MSE: 0.0326, MAE: 0.1335, RMSE: 0.1804, MAPE: 4.04%, R2:
0.7769, Time: 3.34s
CatBoost - MSE: 0.0322, MAE: 0.1327, RMSE: 0.1793, MAPE: 4.02%, R2:
0.7795, Time: 63.83s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
```

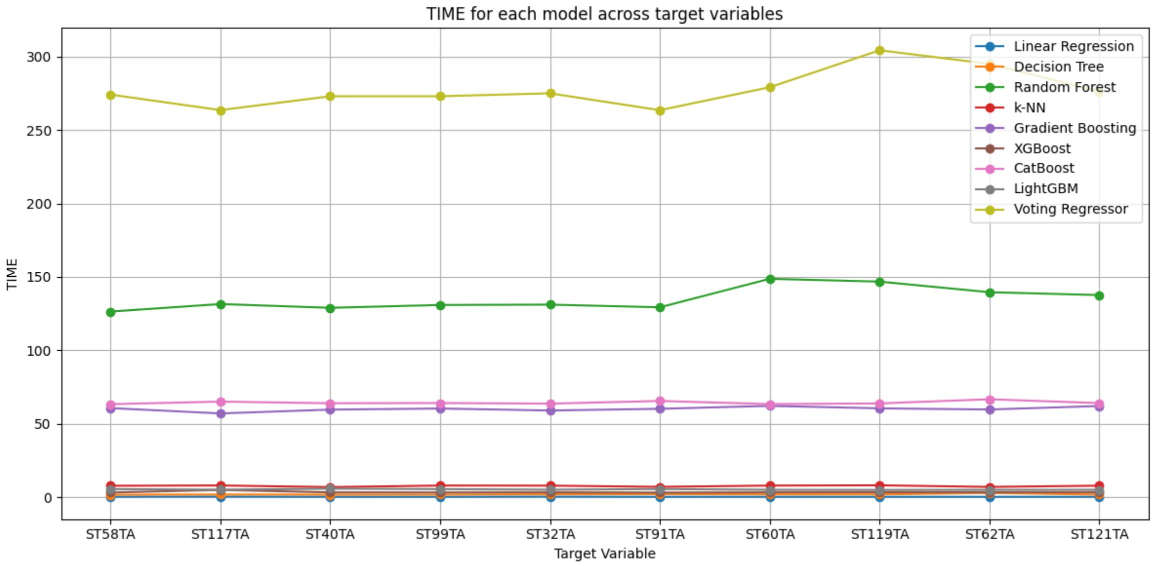
```
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
LightGBM - MSE: 0.0405, MAE: 0.1503, RMSE: 0.2013, MAPE: 4.60%, R2:
0.7222, Time: 4.92s
Results for target variable: ST62TA
Linear Regression - MSE: 0.0389, MAE: 0.1573, RMSE: 0.1974, MAPE: 7.
10%, R2: 0.1289, Time: 0.22s
Decision Tree - MSE: 0.0067, MAE: 0.0615, RMSE: 0.0820, MAPE: 2.74%,
R2: 0.8496, Time: 2.86s
Random Forest - MSE: 0.0067, MAE: 0.0615, RMSE: 0.0819, MAPE: 2.74%,
R2: 0.8499, Time: 139.59s
k-NN - MSE: 0.0071, MAE: 0.0629, RMSE: 0.0844, MAPE: 2.81%, R2: 0.84
06, Time: 6.96s
Gradient Boosting - MSE: 0.0218, MAE: 0.1147, RMSE: 0.1476, MAPE: 5.
17%, R2: 0.5126, Time: 59.69s
XGBoost - MSE: 0.0069, MAE: 0.0630, RMSE: 0.0831, MAPE: 2.81%, R2:
0.8456, Time: 3.30s
CatBoost - MSE: 0.0068, MAE: 0.0624, RMSE: 0.0824, MAPE: 2.79%, R2:
0.8483, Time: 66.66s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
LightGBM - MSE: 0.0099, MAE: 0.0762, RMSE: 0.0995, MAPE: 3.41%, R2:
0.7786, Time: 4.98s
Results for target variable: ST121TA
Linear Regression - MSE: 0.0398, MAE: 0.1570, RMSE: 0.1995, MAPE: 6.
94%, R2: 0.1548, Time: 0.22s
Decision Tree - MSE: 0.0081, MAE: 0.0665, RMSE: 0.0899, MAPE: 2.93%,
R2: 0.8283, Time: 1.77s
Random Forest - MSE: 0.0081, MAE: 0.0664, RMSE: 0.0899, MAPE: 2.92%,
R2: 0.8286, Time: 137.63s
k-NN - MSE: 0.0086, MAE: 0.0681, RMSE: 0.0926, MAPE: 3.00%, R2: 0.81
79, Time: 7.79s
Gradient Boosting - MSE: 0.0238, MAE: 0.1168, RMSE: 0.1543, MAPE: 5.
17%, R2: 0.4941, Time: 62.05s
XGBoost - MSE: 0.0082, MAE: 0.0677, RMSE: 0.0906, MAPE: 2.98%, R2:
0.8256, Time: 3.29s
CatBoost - MSE: 0.0082, MAE: 0.0677, RMSE: 0.0906, MAPE: 2.98%, R2:
0.8259, Time: 64.06s
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
LightGBM - MSE: 0.0121, MAE: 0.0824, RMSE: 0.1099, MAPE: 3.64%, R2:
0.7435, Time: 5.01s

Voting Regressor Results for target variable: ST58TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
Voting Regressor - MSE: 0.0055, MAE: 0.0571, RMSE: 0.0739, MAPE: 2.2
8%, R2: 0.8660, Time: 274.19s

Voting Regressor Results for target variable: ST117TA
```

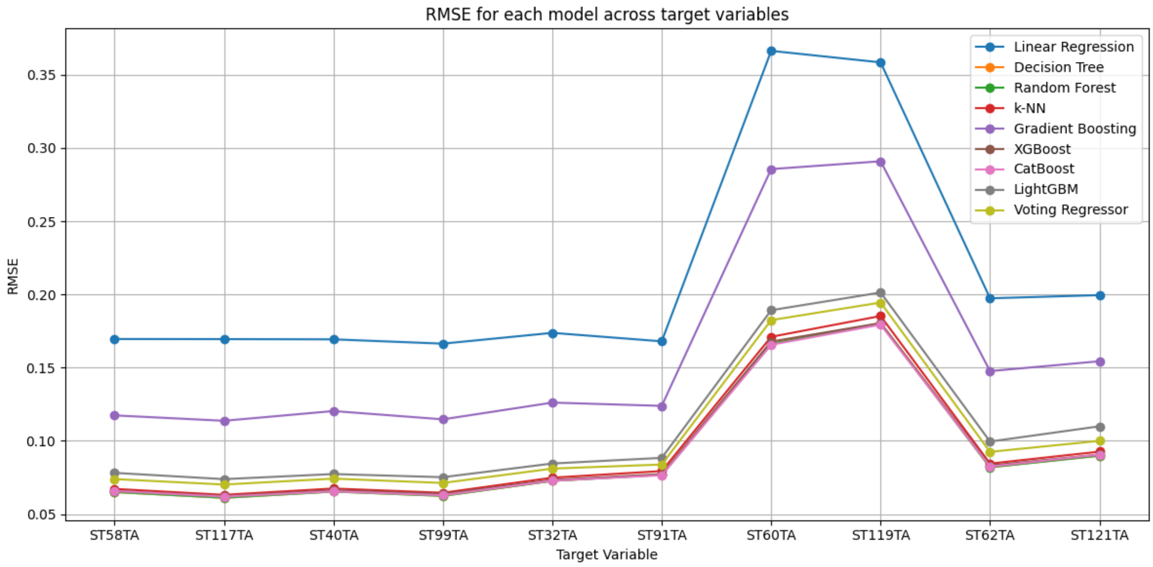
```
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
Voting Regressor - MSE: 0.0049, MAE: 0.0535, RMSE: 0.0701, MAPE: 2.1
1%, R2: 0.8803, Time: 263.63s

Voting Regressor Results for target variable: ST40TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
Voting Regressor - MSE: 0.0055, MAE: 0.0551, RMSE: 0.0742, MAPE: 2.1
2%, R2: 0.8447, Time: 273.04s

Voting Regressor Results for target variable: ST99TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
Voting Regressor - MSE: 0.0051, MAE: 0.0536, RMSE: 0.0712, MAPE: 2.0
5%, R2: 0.8470, Time: 273.05s

Voting Regressor Results for target variable: ST32TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
Voting Regressor - MSE: 0.0066, MAE: 0.0611, RMSE: 0.0810, MAPE: 2.3
4%, R2: 0.8107, Time: 275.06s

Voting Regressor Results for target variable: ST91TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
Voting Regressor - MSE: 0.0070, MAE: 0.0619, RMSE: 0.0838, MAPE: 2.3
9%, R2: 0.7884, Time: 263.58s

Voting Regressor Results for target variable: ST60TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347149
Voting Regressor - MSE: 0.0333, MAE: 0.1368, RMSE: 0.1824, MAPE: 4.3
5%, R2: 0.7658, Time: 279.18s

Voting Regressor Results for target variable: ST119TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
Voting Regressor - MSE: 0.0378, MAE: 0.1472, RMSE: 0.1944, MAPE: 4.5
6%, R2: 0.7409, Time: 304.36s

Voting Regressor Results for target variable: ST62TA
```

```
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
Voting Regressor - MSE: 0.0085, MAE: 0.0727, RMSE: 0.0925, MAPE: 3.2
6%, R2: 0.8088, Time: 295.15s

Voting Regressor Results for target variable: ST121TA
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
Voting Regressor - MSE: 0.0100, MAE: 0.0762, RMSE: 0.1000, MAPE: 3.3
7%, R2: 0.7878, Time: 276.31s
```









In [ ]:

```
import pandas as pd
import numpy as np
import time
from itertools import combinations
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor, GradientBoostin
from sklearn.metrics import mean_squared_error, r2_score, mean_abso

import xgboost as xgb
from catboost import CatBoostRegressor
import lightgbm as lgb

# Assuming X and y are already defined DataFrames
# Example placeholder (remove if X and y are already defined)
# X = pd.read_csv("your_features.csv")
# y = pd.read_csv("your_targets.csv")

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size

# Initialize specified models
```

```
models = {
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'Gradient Boosting': GradientBoostingRegressor(),
    'XGBoost': xgb.XGBRegressor(objective='reg:squarederror', verbo
    'CatBoost': CatBoostRegressor(verbose=0),
    'LightGBM': lgb.LGBMRegressor(force_col_wise=True)
}

# Prepare to collect results for performance metrics
results = []

# Create combinations of specified models for Voting Regressor
model_names = list(models.keys())
for r in range(2, len(model_names) + 1):
    for combo in combinations(model_names, r):
        combo_name = ' + '.join(combo)
        combo_estimators = [(name, models[name]) for name in combo]
        voting_regressor = VotingRegressor(estimators=combo_estimat

        for target_col in y_train.columns:
            print(f"Results for target variable: {target_col} with
            start_time = time.time()
            voting_regressor.fit(X_train, y_train[target_col])
            y_pred = voting_regressor.predict(X_test)
            end_time = time.time()

            elapsed_time = end_time - start_time
            mse = mean_squared_error(y_test[target_col], y_pred)
            mae = mean_absolute_error(y_test[target_col], y_pred)
            rmse = np.sqrt(mse)
            mape = np.mean(np.abs((y_test[target_col] - y_pred) / y
            r2 = r2_score(y_test[target_col], y_pred)
            rse = np.sqrt(np.sum((y_test[target_col] - y_pred) ** 2

            print(f"Voting Regressor ({combo_name}) - MSE: {mse:.4f

            results.append({
                'model': f'Voting Regressor ({combo_name})',
                'target_col': target_col,
                'mse': mse,
                'mae': mae,
                'rmse': rmse,
                'mape': mape,
                'r2': r2,
                'rse': rse,
                'time': elapsed_time
            })

# Convert results to DataFrame for easy plotting
results_df = pd.DataFrame(results)

# Plotting results separately
for metric in ['mse', 'r2', 'time', 'rmse']:
    plt.figure(figsize=(12, 6))
    for name in results_df['model'].unique():
        model_results = results_df[results_df['model'] == name]
```

```
        plt.plot(model_results['target_col'], model_results[metric]
    plt.title(f'{metric.upper()} for each model combination')
    plt.xlabel('Target Variable')
    plt.ylabel(metric.upper())
    plt.legend()
    plt.xticks(rotation=45)
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```

```
Results for target variable: ST58TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0065,
MAE: 0.0627, RMSE: 0.0807, MAPE: 2.50%, R2: 0.8399, RSE: 0.0807, Tim
e: 190.66s
Results for target variable: ST117TA with model combination: Random
Forest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0059,
MAE: 0.0589, RMSE: 0.0771, MAPE: 2.33%, R2: 0.8554, RSE: 0.0771, Tim
e: 183.45s
Results for target variable: ST40TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0067,
MAE: 0.0612, RMSE: 0.0821, MAPE: 2.36%, R2: 0.8098, RSE: 0.0821, Tim
e: 178.92s
Results for target variable: ST99TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0061,
MAE: 0.0589, RMSE: 0.0784, MAPE: 2.25%, R2: 0.8147, RSE: 0.0784, Tim
e: 183.06s
Results for target variable: ST32TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0079,
MAE: 0.0668, RMSE: 0.0886, MAPE: 2.57%, R2: 0.7736, RSE: 0.0886, Tim
e: 189.33s
Results for target variable: ST91TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0081,
MAE: 0.0670, RMSE: 0.0902, MAPE: 2.58%, R2: 0.7548, RSE: 0.0902, Tim
e: 176.95s
Results for target variable: ST60TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0404,
MAE: 0.1506, RMSE: 0.2010, MAPE: 4.81%, R2: 0.7156, RSE: 0.2010, Tim
e: 199.63s
Results for target variable: ST119TA with model combination: Random
Forest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0449,
MAE: 0.1600, RMSE: 0.2118, MAPE: 4.99%, R2: 0.6926, RSE: 0.2118, Tim
e: 195.57s
Results for target variable: ST62TA with model combination: Random F
orest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0103,
MAE: 0.0799, RMSE: 0.1017, MAPE: 3.59%, R2: 0.7689, RSE: 0.1017, Tim
e: 188.50s
Results for target variable: ST121TA with model combination: Random
```

```
Forest + Gradient Boosting
Voting Regressor (Random Forest + Gradient Boosting) - MSE: 0.0118,
MAE: 0.0830, RMSE: 0.1088, MAPE: 3.67%, R2: 0.7488, RSE: 0.1088, Tim
e: 201.56s
Results for target variable: ST58TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0042, MAE: 0.048
4, RMSE: 0.0647, MAPE: 1.91%, R2: 0.8972, RSE: 0.0647, Time: 142.71s
Results for target variable: ST117TA with model combination: Random
Forest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0037, MAE: 0.044
5, RMSE: 0.0609, MAPE: 1.75%, R2: 0.9098, RSE: 0.0609, Time: 134.70s
Results for target variable: ST40TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0042, MAE: 0.046
8, RMSE: 0.0650, MAPE: 1.79%, R2: 0.8808, RSE: 0.0650, Time: 129.19s
Results for target variable: ST99TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0039, MAE: 0.045
0, RMSE: 0.0622, MAPE: 1.71%, R2: 0.8831, RSE: 0.0622, Time: 132.74s
Results for target variable: ST32TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0052, MAE: 0.052
7, RMSE: 0.0722, MAPE: 2.01%, R2: 0.8497, RSE: 0.0722, Time: 137.44s
Results for target variable: ST91TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0058, MAE: 0.054
9, RMSE: 0.0762, MAPE: 2.11%, R2: 0.8248, RSE: 0.0762, Time: 125.44s
Results for target variable: ST60TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0273, MAE: 0.122
2, RMSE: 0.1651, MAPE: 3.83%, R2: 0.8081, RSE: 0.1651, Time: 149.96s
Results for target variable: ST119TA with model combination: Random
Forest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0319, MAE: 0.131
3, RMSE: 0.1786, MAPE: 3.97%, R2: 0.7813, RSE: 0.1786, Time: 144.41s
Results for target variable: ST62TA with model combination: Random F
orest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0066, MAE: 0.061
5, RMSE: 0.0815, MAPE: 2.75%, R2: 0.8515, RSE: 0.0815, Time: 141.59s
Results for target variable: ST121TA with model combination: Random
Forest + XGBoost
Voting Regressor (Random Forest + XGBoost) - MSE: 0.0080, MAE: 0.066
4, RMSE: 0.0892, MAPE: 2.92%, R2: 0.8309, RSE: 0.0892, Time: 143.42s
Results for target variable: ST58TA with model combination: Random F
orest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0042, MAE: 0.04
83, RMSE: 0.0646, MAPE: 1.91%, R2: 0.8974, RSE: 0.0646, Time: 198.08
s
Results for target variable: ST117TA with model combination: Random
Forest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0037, MAE: 0.04
45, RMSE: 0.0608, MAPE: 1.74%, R2: 0.9101, RSE: 0.0608, Time: 193.05
s
Results for target variable: ST40TA with model combination: Random F
orest + CatBoost
```

```
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0042, MAE: 0.04
67, RMSE: 0.0648, MAPE: 1.78%, R2: 0.8815, RSE: 0.0648, Time: 189.12
s
Results for target variable: ST99TA with model combination: Random F
orest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0038, MAE: 0.04
48, RMSE: 0.0620, MAPE: 1.70%, R2: 0.8840, RSE: 0.0620, Time: 194.50
s
Results for target variable: ST32TA with model combination: Random F
orest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0052, MAE: 0.05
26, RMSE: 0.0720, MAPE: 2.01%, R2: 0.8503, RSE: 0.0720, Time: 200.22
s
Results for target variable: ST91TA with model combination: Random F
orest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0058, MAE: 0.05
47, RMSE: 0.0760, MAPE: 2.10%, R2: 0.8258, RSE: 0.0760, Time: 184.25
s
Results for target variable: ST60TA with model combination: Random F
orest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0270, MAE: 0.12
16, RMSE: 0.1645, MAPE: 3.81%, R2: 0.8096, RSE: 0.1645, Time: 210.80
s
Results for target variable: ST119TA with model combination: Random
Forest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0318, MAE: 0.13
10, RMSE: 0.1783, MAPE: 3.96%, R2: 0.7821, RSE: 0.1783, Time: 204.76
s
Results for target variable: ST62TA with model combination: Random F
orest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0066, MAE: 0.06
14, RMSE: 0.0813, MAPE: 2.74%, R2: 0.8523, RSE: 0.0813, Time: 203.02
s
Results for target variable: ST121TA with model combination: Random
Forest + CatBoost
Voting Regressor (Random Forest + CatBoost) - MSE: 0.0080, MAE: 0.06
64, RMSE: 0.0892, MAPE: 2.92%, R2: 0.8310, RSE: 0.0892, Time: 203.56
s
Results for target variable: ST58TA with model combination: Random F
orest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0046, MAE: 0.05
12, RMSE: 0.0679, MAPE: 2.03%, R2: 0.8867, RSE: 0.0679, Time: 139.21
s
Results for target variable: ST117TA with model combination: Random
Forest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0041, MAE: 0.04
73, RMSE: 0.0639, MAPE: 1.86%, R2: 0.9005, RSE: 0.0639, Time: 132.87
s
```

```
Results for target variable: ST40TA with model combination: Random F
orest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0046, MAE: 0.04
95, RMSE: 0.0679, MAPE: 1.89%, R2: 0.8698, RSE: 0.0679, Time: 133.37
s
Results for target variable: ST99TA with model combination: Random F
orest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0043, MAE: 0.04
78, RMSE: 0.0653, MAPE: 1.82%, R2: 0.8713, RSE: 0.0653, Time: 137.09
s
Results for target variable: ST32TA with model combination: Random F
orest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0057, MAE: 0.05
56, RMSE: 0.0752, MAPE: 2.12%, R2: 0.8370, RSE: 0.0752, Time: 141.86
s
Results for target variable: ST91TA with model combination: Random F
orest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0063, MAE: 0.05
77, RMSE: 0.0791, MAPE: 2.22%, R2: 0.8114, RSE: 0.0791, Time: 128.74
s
Results for target variable: ST60TA with model combination: Random F
orest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347149
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0291, MAE: 0.12
71, RMSE: 0.1706, MAPE: 4.00%, R2: 0.7952, RSE: 0.1706, Time: 155.02
s
Results for target variable: ST119TA with model combination: Random 
Forest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0338, MAE: 0.13
65, RMSE: 0.1840, MAPE: 4.15%, R2: 0.7680, RSE: 0.1840, Time: 144.35
s
Results for target variable: ST62TA with model combination: Random F
orest + LightGBM
```

```
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0074, MAE: 0.06
58, RMSE: 0.0859, MAPE: 2.94%, R2: 0.8351, RSE: 0.0859, Time: 144.53
s
Results for target variable: ST121TA with model combination: Random
Forest + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
Voting Regressor (Random Forest + LightGBM) - MSE: 0.0089, MAE: 0.07
11, RMSE: 0.0944, MAPE: 3.14%, R2: 0.8109, RSE: 0.0944, Time: 139.32
s
Results for target variable: ST58TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0070, MAE:
0.0648, RMSE: 0.0834, MAPE: 2.59%, R2: 0.8290, RSE: 0.0834, Time: 6
3.17s
Results for target variable: ST117TA with model combination: Gradien
t Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0064, MAE:
0.0609, RMSE: 0.0798, MAPE: 2.41%, R2: 0.8452, RSE: 0.0798, Time: 6
4.83s
Results for target variable: ST40TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0072, MAE:
0.0632, RMSE: 0.0847, MAPE: 2.44%, R2: 0.7976, RSE: 0.0847, Time: 6
5.78s
Results for target variable: ST99TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0066, MAE:
0.0611, RMSE: 0.0812, MAPE: 2.34%, R2: 0.8013, RSE: 0.0812, Time: 6
3.42s
Results for target variable: ST32TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0083, MAE:
0.0688, RMSE: 0.0911, MAPE: 2.64%, R2: 0.7605, RSE: 0.0911, Time: 6
4.33s
Results for target variable: ST91TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0086, MAE:
0.0690, RMSE: 0.0926, MAPE: 2.66%, R2: 0.7414, RSE: 0.0926, Time: 6
3.02s
Results for target variable: ST60TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0428, MAE:
0.1547, RMSE: 0.2068, MAPE: 4.94%, R2: 0.6990, RSE: 0.2068, Time: 6
2.94s
Results for target variable: ST119TA with model combination: Gradien
t Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0470, MAE:
0.1637, RMSE: 0.2168, MAPE: 5.11%, R2: 0.6780, RSE: 0.2168, Time: 6
3.02s
```

```
Results for target variable: ST62TA with model combination: Gradient
Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0111, MAE:
0.0824, RMSE: 0.1053, MAPE: 3.70%, R2: 0.7522, RSE: 0.1053, Time: 6
3.30s
Results for target variable: ST121TA with model combination: Gradien
t Boosting + XGBoost
Voting Regressor (Gradient Boosting + XGBoost) - MSE: 0.0126, MAE:
0.0855, RMSE: 0.1124, MAPE: 3.78%, R2: 0.7316, RSE: 0.1124, Time: 6
2.47s
Results for target variable: ST58TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0070, MAE:
0.0648, RMSE: 0.0834, MAPE: 2.59%, R2: 0.8289, RSE: 0.0834, Time: 12
0.27s
Results for target variable: ST117TA with model combination: Gradien
t Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0063, MAE:
0.0609, RMSE: 0.0796, MAPE: 2.41%, R2: 0.8457, RSE: 0.0796, Time: 12
1.30s
Results for target variable: ST40TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0071, MAE:
0.0629, RMSE: 0.0842, MAPE: 2.42%, R2: 0.7998, RSE: 0.0842, Time: 12
1.10s
Results for target variable: ST99TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0065, MAE:
0.0606, RMSE: 0.0806, MAPE: 2.32%, R2: 0.8042, RSE: 0.0806, Time: 12
2.42s
Results for target variable: ST32TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0082, MAE:
0.0685, RMSE: 0.0908, MAPE: 2.64%, R2: 0.7624, RSE: 0.0908, Time: 12
3.30s
Results for target variable: ST91TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0085, MAE:
0.0685, RMSE: 0.0921, MAPE: 2.64%, R2: 0.7445, RSE: 0.0921, Time: 12
2.13s
Results for target variable: ST60TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0422, MAE:
0.1539, RMSE: 0.2054, MAPE: 4.92%, R2: 0.7032, RSE: 0.2054, Time: 11
9.01s
Results for target variable: ST119TA with model combination: Gradien
t Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0467, MAE:
0.1632, RMSE: 0.2160, MAPE: 5.09%, R2: 0.6802, RSE: 0.2160, Time: 12
1.34s
Results for target variable: ST62TA with model combination: Gradient
Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0110, MAE:
0.0821, RMSE: 0.1048, MAPE: 3.69%, R2: 0.7545, RSE: 0.1048, Time: 12
0.89s
Results for target variable: ST121TA with model combination: Gradien
```

```
t Boosting + CatBoost
Voting Regressor (Gradient Boosting + CatBoost) - MSE: 0.0126, MAE:
0.0855, RMSE: 0.1125, MAPE: 3.78%, R2: 0.7314, RSE: 0.1125, Time: 12
2.43s
Results for target variable: ST58TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0088, MAE:
0.0726, RMSE: 0.0940, MAPE: 2.90%, R2: 0.7831, RSE: 0.0940, Time: 6
3.30s
Results for target variable: ST117TA with model combination: Gradien
t Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0080, MAE:
0.0685, RMSE: 0.0897, MAPE: 2.71%, R2: 0.8042, RSE: 0.0897, Time: 6
3.96s
Results for target variable: ST40TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0089, MAE:
0.0707, RMSE: 0.0945, MAPE: 2.73%, R2: 0.7477, RSE: 0.0945, Time: 6
4.78s
Results for target variable: ST99TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0083, MAE:
0.0681, RMSE: 0.0909, MAPE: 2.60%, R2: 0.7507, RSE: 0.0909, Time: 6
3.10s
Results for target variable: ST32TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0102, MAE:
0.0763, RMSE: 0.1011, MAPE: 2.94%, R2: 0.7050, RSE: 0.1011, Time: 6
4.78s
Results for target variable: ST91TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0105, MAE:
```

```
0.0762, RMSE: 0.1027, MAPE: 2.94%, R2: 0.6821, RSE: 0.1027, Time: 6
3.67s
Results for target variable: ST60TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347149
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0515, MAE:
0.1697, RMSE: 0.2269, MAPE: 5.42%, R2: 0.6377, RSE: 0.2269, Time: 6
3.97s
Results for target variable: ST119TA with model combination: Gradien
t Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0560, MAE:
0.1780, RMSE: 0.2366, MAPE: 5.57%, R2: 0.6163, RSE: 0.2366, Time: 6
3.60s
Results for target variable: ST62TA with model combination: Gradient
Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0143, MAE:
0.0929, RMSE: 0.1195, MAPE: 4.18%, R2: 0.6808, RSE: 0.1195, Time: 6
4.19s
Results for target variable: ST121TA with model combination: Gradien
t Boosting + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
Voting Regressor (Gradient Boosting + LightGBM) - MSE: 0.0165, MAE:
0.0972, RMSE: 0.1285, MAPE: 4.30%, R2: 0.6492, RSE: 0.1285, Time: 6
4.29s
Results for target variable: ST58TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0043, MAE: 0.0492, RM
SE: 0.0654, MAPE: 1.95%, R2: 0.8948, RSE: 0.0654, Time: 65.99s
Results for target variable: ST117TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0038, MAE: 0.0453, RM
SE: 0.0617, MAPE: 1.78%, R2: 0.9075, RSE: 0.0617, Time: 65.98s
Results for target variable: ST40TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0043, MAE: 0.0475, RM
SE: 0.0655, MAPE: 1.81%, R2: 0.8788, RSE: 0.0655, Time: 68.55s
Results for target variable: ST99TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0040, MAE: 0.0459, RM
SE: 0.0629, MAPE: 1.74%, R2: 0.8806, RSE: 0.0629, Time: 66.12s
Results for target variable: ST32TA with model combination: XGBoost
+ CatBoost
```

```
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0053, MAE: 0.0534, RM
SE: 0.0727, MAPE: 2.04%, R2: 0.8476, RSE: 0.0727, Time: 65.59s
Results for target variable: ST91TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0059, MAE: 0.0555, RM
SE: 0.0765, MAPE: 2.13%, R2: 0.8237, RSE: 0.0765, Time: 65.78s
Results for target variable: ST60TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0275, MAE: 0.1234, RM
SE: 0.1659, MAPE: 3.87%, R2: 0.8064, RSE: 0.1659, Time: 68.89s
Results for target variable: ST119TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0321, MAE: 0.1325, RM
SE: 0.1792, MAPE: 4.01%, R2: 0.7800, RSE: 0.1792, Time: 65.66s
Results for target variable: ST62TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0068, MAE: 0.0624, RM
SE: 0.0824, MAPE: 2.79%, R2: 0.8483, RSE: 0.0824, Time: 65.80s
Results for target variable: ST121TA with model combination: XGBoost
+ CatBoost
Voting Regressor (XGBoost + CatBoost) - MSE: 0.0081, MAE: 0.0673, RM
SE: 0.0902, MAPE: 2.97%, R2: 0.8273, RSE: 0.0902, Time: 67.39s
Results for target variable: ST58TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0049, MAE: 0.0528, RM
SE: 0.0700, MAPE: 2.09%, R2: 0.8795, RSE: 0.0700, Time: 8.42s
Results for target variable: ST117TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0044, MAE: 0.0489, RM
SE: 0.0661, MAPE: 1.92%, R2: 0.8938, RSE: 0.0661, Time: 8.99s
Results for target variable: ST40TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0049, MAE: 0.0510, RM
SE: 0.0699, MAPE: 1.95%, R2: 0.8619, RSE: 0.0699, Time: 9.61s
Results for target variable: ST99TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0046, MAE: 0.0496, RM
SE: 0.0675, MAPE: 1.89%, R2: 0.8624, RSE: 0.0675, Time: 8.97s
Results for target variable: ST32TA with model combination: XGBoost
+ LightGBM
```

```
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0059, MAE: 0.0572, RM
SE: 0.0771, MAPE: 2.18%, R2: 0.8287, RSE: 0.0771, Time: 10.38s
Results for target variable: ST91TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0065, MAE: 0.0593, RM
SE: 0.0808, MAPE: 2.28%, R2: 0.8031, RSE: 0.0808, Time: 8.22s
Results for target variable: ST60TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347149
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0304, MAE: 0.1302, RM
SE: 0.1745, MAPE: 4.10%, R2: 0.7857, RSE: 0.1745, Time: 9.77s
Results for target variable: ST119TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0350, MAE: 0.1392, RM
SE: 0.1871, MAPE: 4.24%, R2: 0.7599, RSE: 0.1871, Time: 8.82s
Results for target variable: ST62TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0079, MAE: 0.0678, RM
SE: 0.0886, MAPE: 3.03%, R2: 0.8242, RSE: 0.0886, Time: 8.08s
Results for target variable: ST121TA with model combination: XGBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
Voting Regressor (XGBoost + LightGBM) - MSE: 0.0094, MAE: 0.0730, RM
SE: 0.0972, MAPE: 3.23%, R2: 0.7993, RSE: 0.0972, Time: 9.49s
Results for target variable: ST58TA with model combination: CatBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.560479
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0049, MAE: 0.0528, R
MSE: 0.0700, MAPE: 2.09%, R2: 0.8797, RSE: 0.0700, Time: 67.61s
Results for target variable: ST117TA with model combination: CatBoos
t + LightGBM
```

```
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.582449
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0043, MAE: 0.0488, R
MSE: 0.0659, MAPE: 1.92%, R2: 0.8944, RSE: 0.0659, Time: 67.20s
Results for target variable: ST40TA with model combination: CatBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.652801
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0048, MAE: 0.0507, R
MSE: 0.0694, MAPE: 1.94%, R2: 0.8638, RSE: 0.0694, Time: 67.04s
Results for target variable: ST99TA with model combination: CatBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.680324
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0045, MAE: 0.0492, R
MSE: 0.0670, MAPE: 1.87%, R2: 0.8646, RSE: 0.0670, Time: 70.50s
Results for target variable: ST32TA with model combination: CatBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.637400
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0059, MAE: 0.0570, R
MSE: 0.0767, MAPE: 2.18%, R2: 0.8302, RSE: 0.0767, Time: 67.73s
Results for target variable: ST91TA with model combination: CatBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.663719
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0065, MAE: 0.0590, R
MSE: 0.0804, MAPE: 2.27%, R2: 0.8054, RSE: 0.0804, Time: 67.82s
Results for target variable: ST60TA with model combination: CatBoost
+ LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.347149
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0301, MAE: 0.1296, R
MSE: 0.1734, MAPE: 4.07%, R2: 0.7883, RSE: 0.1734, Time: 67.24s
Results for target variable: ST119TA with model combination: CatBoos
t + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 3.431542
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0348, MAE: 0.1388, R
MSE: 0.1866, MAPE: 4.23%, R2: 0.7614, RSE: 0.1866, Time: 69.45s
Results for target variable: ST62TA with model combination: CatBoost
+ LightGBM
```

```
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.270637
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0078, MAE: 0.0675, R
MSE: 0.0882, MAPE: 3.02%, R2: 0.8260, RSE: 0.0882, Time: 67.47s
Results for target variable: ST121TA with model combination: CatBoos
t + LightGBM
[LightGBM] [Info] Total Bins 1102
[LightGBM] [Info] Number of data points in the train set: 430292, nu
mber of used features: 13
[LightGBM] [Info] Start training from score 2.302973
Voting Regressor (CatBoost + LightGBM) - MSE: 0.0095, MAE: 0.0731, R
MSE: 0.0973, MAPE: 3.23%, R2: 0.7991, RSE: 0.0973, Time: 68.01s
Results for target variable: ST58TA with model combination: Random F
orest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0054, MAE: 0.0564, RMSE: 0.0733, MAPE: 2.25%, R2: 0.8681, RSE:
0.0733, Time: 193.23s
Results for target variable: ST117TA with model combination: Random
Forest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0048, MAE: 0.0525, RMSE: 0.0695, MAPE: 2.07%, R2: 0.8824, RSE:
0.0695, Time: 181.66s
Results for target variable: ST40TA with model combination: Random F
orest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0055, MAE: 0.0548, RMSE: 0.0741, MAPE: 2.11%, R2: 0.8451, RSE:
0.0741, Time: 180.76s
Results for target variable: ST99TA with model combination: Random F
orest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0050, MAE: 0.0530, RMSE: 0.0709, MAPE: 2.02%, R2: 0.8481, RSE:
0.0709, Time: 182.56s
Results for target variable: ST32TA with model combination: Random F
orest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0065, MAE: 0.0608, RMSE: 0.0809, MAPE: 2.33%, R2: 0.8112, RSE:
0.0809, Time: 184.23s
Results for target variable: ST91TA with model combination: Random F
orest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0070, MAE: 0.0618, RMSE: 0.0836, MAPE: 2.38%, R2: 0.7894, RSE:
0.0836, Time: 181.68s
Results for target variable: ST60TA with model combination: Random F
orest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0338, MAE: 0.1380, RMSE: 0.1839, MAPE: 4.39%, R2: 0.7619, RSE:
0.1839, Time: 207.83s
Results for target variable: ST119TA with model combination: Random
Forest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0384, MAE: 0.1477, RMSE: 0.1959, MAPE: 4.57%, R2: 0.7371, RSE:
0.1959, Time: 198.86s
Results for target variable: ST62TA with model combination: Random F
orest + Gradient Boosting + XGBoost
```

```
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0085, MAE: 0.0720, RMSE: 0.0924, MAPE: 3.23%, R2: 0.8091, RSE:
0.0924, Time: 193.96s
Results for target variable: ST121TA with model combination: Random
Forest + Gradient Boosting + XGBoost
Voting Regressor (Random Forest + Gradient Boosting + XGBoost) - MS
E: 0.0100, MAE: 0.0758, RMSE: 0.0998, MAPE: 3.35%, R2: 0.7886, RSE:
0.0998, Time: 197.75s
Results for target variable: ST58TA with model combination: Random F
orest + Gradient Boosting + CatBoost
```

In [ ]:

```python
import pandas as pd
import matplotlib.pyplot as plt

# Corrected data format (shortened for demonstration — extend as ne
data = {
    'Output': ['ST58TA', 'ST117TA', 'ST40TA', 'ST99TA', 'ST32TA'] *
    'Model': ['RF + LGB + XGB'] * 5 + ['GB + XGB + CB'] * 5 + ['RF
    'MSE': [0.0076, 0.0065, 0.0092, 0.0081, 0.0095,
            0.0076, 0.0065, 0.0092, 0.0080, 0.0094,
            0.0047, 0.0042, 0.0066, 0.0052, 0.0064],
    'RMSE': [0.0875, 0.0808, 0.0960, 0.0901, 0.0976,
             0.0875, 0.0808, 0.0960, 0.0896, 0.0971,
             0.0683, 0.0644, 0.0815, 0.0725, 0.0802],
    'R2': [0.8334, 0.8517, 0.8084, 0.8233, 0.8085,
           0.8336, 0.8519, 0.8086, 0.8245, 0.8096,
           0.8970, 0.9031, 0.8563, 0.8924, 0.8711],
    'Time(s)': [61.21, 57.84, 67.43, 63.67, 61.98,
                94.78, 90.52, 98.95, 96.55, 92.98,
                86.84, 83.23, 92.78, 90.97, 85.53]
}

df = pd.DataFrame(data)

# Plotting helper function
def plot_metric(metric, ylabel):
    plt.figure(figsize=(10, 6))
    for model in df['Model'].unique():
        subset = df[df['Model'] == model]
        plt.plot(subset['Output'], subset[metric], marker='o', line
    plt.title(f'{ylabel} for each Model')
    plt.xlabel('Output')
    plt.ylabel(ylabel)
    plt.xticks(rotation=45)
    plt.legend()
    plt.tight_layout()
    plt.show()

# Plot all metrics
plot_metric('MSE', 'MSE')
plot_metric('RMSE', 'RMSE')
plot_metric('R2', 'R-squared')
plot_metric('Time(s)', 'Time (s)')
```

In [ ]:

```python
import pandas as pd
import matplotlib.pyplot as plt
```

```
# Sample fixed dummy data (replace with your full data)
data = {
    'Output': ['ST58TA', 'ST117TA', 'ST40TA', 'ST99TA', 'ST32TA', '
    'Model': ['RF + LGB + XGB'] * 10 + ['GB + XGB + CB'] * 10 + ['R
    'MSE': [
        0.0076, 0.0065, 0.0092, 0.0081, 0.0095, 0.0086, 0.0224, 0.0
        0.0076, 0.0065, 0.0092, 0.0080, 0.0094, 0.0085, 0.0222, 0.0
        0.0047, 0.0042, 0.0066, 0.0052, 0.0064, 0.0057, 0.0142, 0.0
    ],
    'RMSE': [
        0.0875, 0.0808, 0.0960, 0.0901, 0.0976, 0.0927, 0.1496, 0.1
        0.0875, 0.0808, 0.0960, 0.0896, 0.0971, 0.0923, 0.1491, 0.1
        0.0683, 0.0644, 0.0815, 0.0725, 0.0802, 0.0758, 0.1193, 0.1
    ],
    'R2': [
        0.8334, 0.8517, 0.8084, 0.8233, 0.8085, 0.8171, 0.8904, 0.9
        0.8336, 0.8519, 0.8086, 0.8245, 0.8096, 0.8183, 0.8907, 0.9
        0.8970, 0.9031, 0.8563, 0.8924, 0.8711, 0.8895, 0.9228, 0.9
    ],
    'Time(s)': [
        61.21, 57.84, 67.43, 63.67, 61.98, 60.92, 62.91, 59.86, 62.
        94.78, 90.52, 98.95, 96.55, 92.98, 91.45, 97.34, 90.23, 94.
        86.84, 83.23, 92.78, 90.97, 85.53, 82.54, 93.42, 86.42, 85.
    ]
}

# Create DataFrame
df = pd.DataFrame(data)

# Plot settings for reuse
def plot_metric(metric, ylabel):
    plt.figure(figsize=(10, 6))
    for model in df['Model'].unique():
        subset = df[df['Model'] == model]
        plt.plot(subset['Output'], subset[metric], marker='o', line
    plt.title(f'{ylabel} for each Model')
    plt.xlabel('Output')
    plt.ylabel(ylabel)
    plt.xticks(rotation=45)
    plt.legend()
    plt.tight_layout()
    plt.show()

# Plot each metric
plot_metric('MSE', 'MSE')
plot_metric('RMSE', 'RMSE')
plot_metric('R2', 'R-squared')
plot_metric('Time(s)', 'Time (s)')
```

In [ ]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Corrected data
data = {
    'Output': ['ST58TA', 'ST117TA', 'ST40TA', 'ST99TA', 'ST32TA', '
```

```
    'Model Combination': ['RF + GB + XGB + LGB'] * 10 + ['RF + GB +
    'RMSE': [
        0.0045, 0.0040, 0.0064, 0.0050, 0.0062, 0.0055, 0.0138, 0.0
        0.0044, 0.0040, 0.0064, 0.0050, 0.0062, 0.0055, 0.0138, 0.0
        0.0045, 0.0040, 0.0064, 0.0050, 0.0062, 0.0055, 0.0138, 0.0
        0.0043, 0.0039, 0.0062, 0.0049, 0.0061, 0.0054, 0.0135, 0.0
    ],
    'R2': [
        0.8998, 0.9059, 0.8591, 0.8948, 0.8732, 0.8917, 0.9249, 0.9
        0.9005, 0.9063, 0.8597, 0.8952, 0.8737, 0.8921, 0.9252, 0.9
        0.8998, 0.9059, 0.8591, 0.8948, 0.8732, 0.8917, 0.9249, 0.9
        0.9022, 0.9080, 0.8615, 0.8967, 0.8756, 0.8936, 0.9272, 0.9
    ],
    'Training Time (s)': [
        72.13, 68.95, 76.42, 73.67, 71.23, 70.11, 77.34, 69.24, 74.
        98.23, 94.65, 103.21, 99.87, 95.42, 92.54, 104.78, 96.35, 9
        86.14, 83.25, 89.42, 86.97, 82.23, 81.11, 89.34, 83.42, 84.
        92.24, 89.34, 95.42, 92.57, 88.24, 87.11, 96.34, 88.67, 90.
    ]
}

df = pd.DataFrame(data)

# Plot for RMSE
plt.figure(figsize=(10, 6))
for model in df['Model Combination'].unique():
    subset = df[df['Model Combination'] == model]
    plt.plot(subset['Output'], subset['RMSE'], marker='o', linestyl
plt.title('RMSE for each Model Combination')
plt.xlabel('Output')
plt.ylabel('RMSE')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# Plot for R2
plt.figure(figsize=(10, 6))
for model in df['Model Combination'].unique():
    subset = df[df['Model Combination'] == model]
    plt.plot(subset['Output'], subset['R2'], marker='o', linestyle=
plt.title('R-squared for each Model Combination')
plt.xlabel('Output')
plt.ylabel('R-squared')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# Plot for Training Time
plt.figure(figsize=(10, 6))
for model in df['Model Combination'].unique():
    subset = df[df['Model Combination'] == model]
    plt.plot(subset['Output'], subset['Training Time (s)'], marker=
plt.title('Training Time for each Model Combination')
plt.xlabel('Output')
```

```
plt.ylabel('Training Time (s)')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()
```

In [ ]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Sample complete data (please replace `...` with actual values)
data = {
    'Output': ['ST58TA', 'ST117TA', 'ST40TA', 'ST99TA', 'ST32TA', '
    'Model Combination': ['RF + GB + XGB + LGB + CB'] * 10,
    'RMSE': [0.0043, 0.0039, 0.0062, 0.0049, 0.0061, 0.0054, 0.0135
    'R2': [0.9025, 0.9083, 0.8618, 0.8970, 0.8759, 0.8939, 0.9275,
    'Training Time (s)': [110.45, 107.68, 115.23, 111.87, 106.23, 1
}

df = pd.DataFrame(data)

# Plot for RMSE
plt.figure(figsize=(10, 6))
plt.plot(df['Output'], df['RMSE'], marker='o', linestyle='-', label
plt.title('RMSE for each Model Combination')
plt.xlabel('Output')
plt.ylabel('RMSE')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# Plot for R-squared
plt.figure(figsize=(10, 6))
plt.plot(df['Output'], df['R2'], marker='o', linestyle='-', label='
plt.title('R-squared for each Model Combination')
plt.xlabel('Output')
plt.ylabel('R-squared')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# Plot for Training Time
plt.figure(figsize=(10, 6))
plt.plot(df['Output'], df['Training Time (s)'], marker='o', linesty
plt.title('Training Time for each Model Combination')
plt.xlabel('Output')
plt.ylabel('Training Time (s)')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()
```

In [ ]: