We used TPOT(Tree Based Pipeline Optimisation Tool) to automate our model selection and hyperparameter optimisation resulting in random forest regressor for air quality analysis for our dataset.

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

#Prediction model for PM10
X = final_data[['Hour', 'Day', 'Month', 'Year', 'CO', 'Season']]
y = final_data['PM10']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
print(X_train.shape)
```

```
(23144, 6)
```

```
print(X_test.shape)
```

```
(5787, 6)
```

```
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10, 15],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

```
from sklearn.model_selection import RandomizedSearchCV
random = RandomizedSearchCV(estimator=RandomForestRegressor(),param_distributions=param_grid,n_iter=10,cv=3,verbose=2,n_jobs=-1)
```

```
random.fit(X_train_scaled,y_train)
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
▸ RandomizedSearchCV
▸ best_estimator_: RandomForestRegressor
▸ RandomForestRegressor
```

```
model_param = random.best_params_
```

```
model_param
```

```
{'n_estimators': 300,
 'min_samples_split': 10,
 'min_samples_leaf': 2,
 'max_depth': None}
```

```
forest = RandomForestRegressor(n_estimators=300,min_samples_split=10,min_samples_leaf=2,max_depth=None)
```

```
forest.fit(X_train_scaled,y_train)
```

```
▾ RandomForestRegressor
RandomForestRegressor(min_samples_leaf=2, min_samples_split=10,
                      n_estimators=300)
```

```
y_pred = forest.predict(X_test_scaled)
y_pred
```

```
array([ 84.10395147, 282.40259937, 195.9675831 , ..., 307.78612972,
        62.25392029, 277.71095366])
```

```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("RMSE: ",np.sqrt(mse))
print(f'R-squared: {r2}')
```

```
RMSE:  47.33612806395524
R-squared: 0.7778909217614169
```

```
accuracy = forest.score(X_test_scaled,y_test)
print(accuracy)
```

```
0.7778909217614169
```

```
#Scaling for the PM10
scaler = StandardScaler()
X2_train_scaled = scaler.fit_transform(X2_train)
X2_test_scaled = scaler.transform(X2_test)
```

```
random.fit(X2_train_scaled,z_train)
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

▸ **RandomizedSearchCV**

▸ **best_estimator_: RandomForestRegressor**

▸ RandomForestRegressor