Step 1: Data Preparation

```
from google.colab import drive
```

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import pandas as pd

data_dir = '/content/drive/MyDrive/Fire Detection'
```

```
excel_path = f'{data_dir}/Field Measurements.xlsx'
df = pd.read_excel(excel_path)
print(df.head())
```

```
           Unnamed: 0  MOL_WGHT_CALC  SOUND_SPEED   FLOW_RATE  \
0 2022-10-05 00:00:00      20.853416   389.122253   51.550770   
1 2022-10-05 00:01:00      20.879612   388.965332   59.074341   
2 2022-10-05 00:02:00      20.879612   388.905548  144.348465   
3 2022-10-05 00:03:00      20.879612   389.000061   65.950691   
4 2022-10-05 00:04:00      20.895794   388.885010  150.639832   

   FLARE_MASS_FLOW_RATE_FROM_FLOW_METER  STM_FLOW_RATE  FUEL_GAS_DMND  \
0                              1.002490     545.469727      35.769882   
1                              1.130493     517.915405      41.197296   
2                              2.762231     508.212128      52.288105   
3                              1.262085     502.861816      40.961323   
4                              2.896216     513.470154      39.545475   

   STEAM_DEMAND  HEAT_VAL_MOL  PRESSURE  ...  Carbon Dioxide Emissions  \
0    278.895691      5.519865  0.008345  ...                167.949142   
1    256.278748      5.964423  0.008346  ...                186.606491   
2    267.592102      2.862000  0.008347  ...                  0.000000   
3    268.124329      8.109967  0.008348  ...                345.279266   
4    276.079559      5.529770  0.008349  ...                163.863922   

   CO2-Equivalent Emissions based on Global Warming Potential  \
0                                         632.421997            
1                                         545.707642            
2                                         759.017273            
3                                         501.124146            
4                                         638.406677            
```

```
   Volatile Organic Compound Emissions       DRE        CE  \
0                            26.824116  0.710643  0.658651   
1                            20.781002  0.776066  0.731054   
2                            43.936718  0.000000  0.000000   
3                             9.014403  0.934150  0.912100   
4                            27.493492  0.700933  0.648023   

   N2 Flow downstream of Flare flow meter  \
0                                       0   
1                                       0   
2                                       0   
3                                       0   
4                                       0   

   Double-bond hydrocarbon background, Percentage  \
0                                               0   
1                                               0   
2                                               0   
3                                               0   
4                                               0   

   Ratio of Carbon atoms to Hydrogen atoms  Cross-wind-speed  Unnamed: 28  
0                                     0.25          1.405809          NaN  
1                                     0.25          1.054357          NaN  
2                                     0.25          1.252845          NaN  
3                                     0.25          0.966949          NaN  
4                                     0.25          1.737230          NaN  

[5 rows x 29 columns]
```

```
import pandas as pd

# Load the Excel file
excel_path = '/content/drive/MyDrive/Fire Detection/cleaned_data.xlsx'  # Adjust
df = pd.read_excel(excel_path)

# Step 1: Ensure 'Unnamed: 0' is treated as a datetime object and rename it to 'D
df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'])
df.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)

# Step 2: Filter Data Range (13:15 to 16:15)
start_time = '2022-10-05 13:15:00'
end_time = '2022-10-05 16:15:00'
df_filtered = df[(df['Date'] >= start_time) & (df['Date'] <= end_time)]

# Step 3: Remove rows where CE or DRE values are zero
df_filtered = df_filtered[(df_filtered['CE'] != 0) & (df_filtered['DRE'] != 0)]

# Step 4: Remove columns with constant values
constant_columns = [col for col in df_filtered.columns if df_filtered[col].nuniqu
df_filtered = df_filtered.drop(columns=constant_columns)

# Step 5: Drop columns with text data (non-numeric types other than date)
non_numeric_columns = df_filtered.select_dtypes(include=['object']).columns
df_filtered = df_filtered.drop(columns=non_numeric_columns)

# Save the cleaned data to a new Excel file
output_excel_path = '/content/drive/MyDrive/Fire Detection/cleaned_data.xlsx'
df_filtered.to_excel(output_excel_path, index=False)

print(f"Filtered data saved to {output_excel_path}")
```

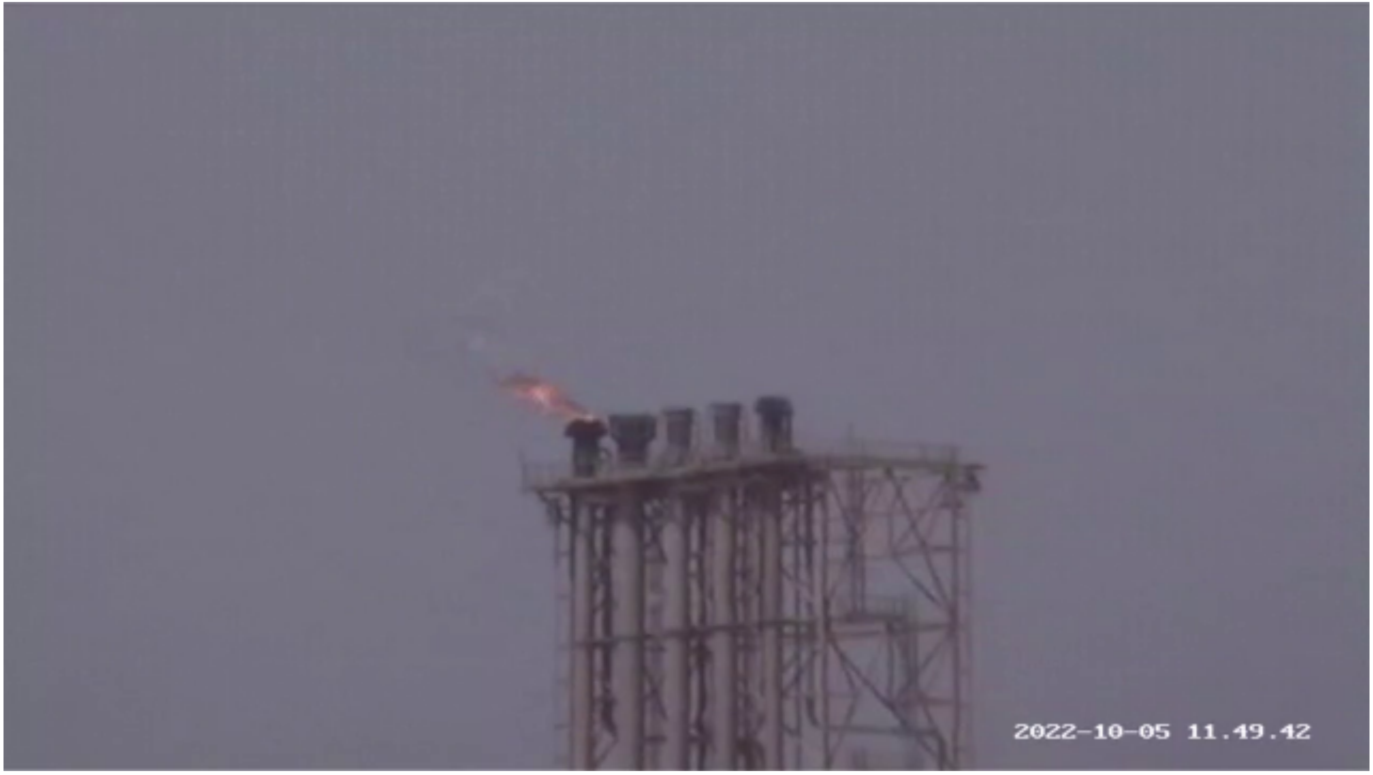
```
Filtered data saved to /content/drive/MyDrive/Fire Detection/cleaned_data.xls
```

Step 2: Video Frame Extraction

```
import cv2
from google.colab.patches import cv2_imshow
import os
video_path = f'{data_dir}/Fire_video.mkv'

cap = cv2.VideoCapture(video_path)
```

```
if not cap.isOpened():
    print("Error: Could not open video file.")
else:
    print("Video file opened successfully.")

    frame_count = 0
    target_frame = 1000  # The frame number we want to display

    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            print("Reached the end of the video or encountered an error.")
            break
        frame_count += 1
        if frame_count == target_frame:

            resized_frame = cv2.resize(frame, (640, 360))

            cv2_imshow(resized_frame)

            break

cap.release()
```

```
Video file opened successfully.
```



```
import cv2
import os

def extract_specific_second_each_minute(video_path, output_folder, start_time="13
    os.makedirs(output_folder, exist_ok=True)

    cap = cv2.VideoCapture(video_path)
    if not cap.isOpened():
        print(f"Error: Could not open video file: {video_path}")
        return

    video_fps = cap.get(cv2.CAP_PROP_FPS)
    total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    print(f"Video FPS: {video_fps}, Total Frames: {total_frames}")

    # Helper function to convert time to seconds
    def time_to_seconds(time_str):
        h, m, s = map(float, time_str.split(":"))
        return h * 3600 + m * 60 + s

    start_seconds = time_to_seconds(start_time)
    end_seconds = time_to_seconds(end_time)
    # Target second within each minute to extract frames
    target_second = int(start_seconds % 60)
    current_time = start_seconds

    while current_time <= end_seconds:
        # Calculate the start of the current minute and the target second within
        current_minute_start = (current_time // 60) * 60
        target_time_in_current_minute = current_minute_start + target_second

        if target_time_in_current_minute > end_seconds:
            break

        # Frame numbers for the 1st and 30th frames within that second
        frame_number_start = int(target_time_in_current_minute * video_fps)
        frame_number_30 = frame_number_start + 29  # 30th frame

        # Extract the first frame
        cap.set(cv2.CAP_PROP_POS_FRAMES, frame_number_start)
        ret, frame = cap.read()
        if ret:
            time_str = f'{int(target_time_in_current_minute // 3600):02d}:{int((t
            output_path = os.path.join(output_folder, f'frame_{time_str}_first.jp
            cv2.imwrite(output_path, frame)
            print(f"Extracted first frame from {time_str}")
        else:
            print(f"Error reading first frame at {time_str}")

        # Extract the 30th frame if it exists
        if frame_number_30 < total_frames:
            cap.set(cv2.CAP_PROP_POS_FRAMES, frame_number_30)
            ret, frame = cap.read()
            if ret:
                output_path = os.path.join(output_folder, f'frame_{time_str}_last
                cv2.imwrite(output_path, frame)
                print(f"Extracted last frame from {time_str}")
            else:
                print(f"Error reading last frame at {time_str}")

        # Move to the next minute
        current_time = (current_time // 60 + 1) * 60

    cap.release()
    print("Frame extraction completed.")

# Specify input and output paths
```

```
input_video_path = '/content/drive/MyDrive/Fire Detection/Fire_video.mkv'
output_folder = '/content/drive/MyDrive/Fire Detection/extracted_frames'

# Run the extraction function
extract_specific_second_each_minute(input_video_path, output_folder, start_time="
```

```
Video FPS: 60.0, Total Frames: 2059085
Extracted first frame from 01:25:34
Extracted last frame from 01:25:34
Extracted first frame from 01:26:34
Extracted last frame from 01:26:34
Extracted first frame from 01:27:34
Extracted last frame from 01:27:34
Extracted first frame from 01:28:34
Extracted last frame from 01:28:34
Extracted first frame from 01:29:34
Extracted last frame from 01:29:34
Extracted first frame from 01:30:34
Extracted last frame from 01:30:34
Extracted first frame from 01:31:34
Extracted last frame from 01:31:34
Extracted first frame from 01:32:34
Extracted last frame from 01:32:34
Extracted first frame from 01:33:34
Extracted last frame from 01:33:34
Extracted first frame from 01:34:34
Extracted last frame from 01:34:34
Extracted first frame from 01:35:34
Extracted last frame from 01:35:34
Extracted first frame from 01:36:34
Extracted last frame from 01:36:34
Extracted first frame from 01:37:34
Extracted last frame from 01:37:34
Extracted first frame from 01:38:34
Extracted last frame from 01:38:34
Extracted first frame from 01:39:34
Extracted last frame from 01:39:34
Extracted first frame from 01:40:34
Extracted last frame from 01:40:34
Extracted first frame from 01:41:34
Extracted last frame from 01:41:34
Extracted first frame from 01:42:34
Extracted last frame from 01:42:34
Extracted first frame from 01:43:34
Extracted last frame from 01:43:34
Extracted first frame from 01:44:34
Extracted last frame from 01:44:34
Extracted first frame from 01:45:34
Extracted last frame from 01:45:34
Extracted last frame from 01:45:34
Extracted first frame from 01:46:34
Extracted last frame from 01:46:34
Extracted first frame from 01:47:34
Extracted last frame from 01:47:34
Extracted first frame from 01:48:34
Extracted last frame from 01:48:34
Extracted first frame from 01:49:34
Extracted last frame from 01:49:34
Extracted first frame from 01:50:34
Extracted last frame from 01:50:34
Extracted first frame from 01:51:34
Extracted last frame from 01:51:34
Extracted first frame from 01:52:34
Extracted last frame from 01:52:34
Extracted first frame from 01:53:34
Extracted last frame from 01:53:34
Extracted first frame from 01:54:34
```

```
import cv2
import os
from datetime import datetime, timedelta

def extract_specific_second_each_minute(video_path, output_folder, video_start_ti
    os.makedirs(output_folder, exist_ok=True)
    cap = cv2.VideoCapture(video_path)
    if not cap.isOpened():
        print(f"Error: Could not open video file: {video_path}")
        return
    video_fps = cap.get(cv2.CAP_PROP_FPS)
    total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    def time_to_seconds(time_str):
        h, m, s = map(float, time_str.split(":"))
        return h * 3600 + m * 60 + s
    start_seconds = time_to_seconds(video_start_time)
    end_seconds = time_to_seconds(end_time)
    video_start_dt = datetime.strptime(video_start_time, "%H:%M:%S")
    desired_start_dt = datetime.strptime(desired_start_time, "%H:%M:%S")
    time_offset = desired_start_dt - video_start_dt
    target_second = int(start_seconds % 60)
    current_time = start_seconds
    while current_time <= end_seconds:
        current_minute_start = (current_time // 60) * 60
        target_time_in_current_minute = current_minute_start + target_second
        if target_time_in_current_minute > end_seconds:
            break
        frame_number_start = int(target_time_in_current_minute * video_fps)
        frame_number_30 = frame_number_start + 29
        adjusted_time_str = (datetime.strptime("00:00:00", "%H:%M:%S") + timedelt
        cap.set(cv2.CAP_PROP_POS_FRAMES, frame_number_start)
        ret, frame = cap.read()
        if ret:
            output_path = os.path.join(output_folder, f'frame_{adjusted_time_str}
            cv2.imwrite(output_path, frame)
            print(f"Extracted first frame from {adjusted_time_str}")
        else:
            print(f"Error reading first frame at {adjusted_time_str}")
        if frame_number_30 < total_frames:
            cap.set(cv2.CAP_PROP_POS_FRAMES, frame_number_30)
            ret, frame = cap.read()
            if ret:
                output_path = os.path.join(output_folder, f'frame_{adjusted_time_
                cv2.imwrite(output_path, frame)
                print(f"Extracted last frame from {adjusted_time_str}")
            else:
                print(f"Error reading last frame at {adjusted_time_str}")
        current_time = (current_time // 60 + 1) * 60
    cap.release()
    print("Frame extraction completed.")

input_video_path = '/content/drive/MyDrive/Fire Detection/Fire_video.mkv'
output_folder = '/content/drive/MyDrive/Fire Detection/new_extracted_frames'
extract_specific_second_each_minute(input_video_path, output_folder, video_start_
```

```
Extracted first frame from 13:15:00
Extracted last frame from 13:15:00
Extracted first frame from 13:16:00
Extracted last frame from 13:16:00
Extracted first frame from 13:17:00
Extracted last frame from 13:17:00
Extracted first frame from 13:18:00
Extracted last frame from 13:18:00
Extracted first frame from 13:19:00
Extracted last frame from 13:19:00
Extracted first frame from 13:20:00
Extracted last frame from 13:20:00
Extracted first frame from 13:21:00
Extracted last frame from 13:21:00
Extracted first frame from 13:22:00
Extracted last frame from 13:22:00
Extracted first frame from 13:23:00
Extracted last frame from 13:23:00
Extracted first frame from 13:24:00
Extracted last frame from 13:24:00
Extracted first frame from 13:25:00
Extracted last frame from 13:25:00
Extracted first frame from 13:26:00
Extracted last frame from 13:26:00
Extracted first frame from 13:27:00
Extracted last frame from 13:27:00
Extracted first frame from 13:28:00
Extracted last frame from 13:28:00
Extracted first frame from 13:29:00
Extracted last frame from 13:29:00
Extracted first frame from 13:30:00
Extracted last frame from 13:30:00
Extracted first frame from 13:31:00
Extracted last frame from 13:31:00
Extracted first frame from 13:32:00
Extracted last frame from 13:32:00
Extracted first frame from 13:33:00
Extracted last frame from 13:33:00
Extracted first frame from 13:34:00
Extracted last frame from 13:34:00
Extracted first frame from 13:35:00
Extracted last frame from 13:35:00
Extracted first frame from 13:36:00
Extracted last frame from 13:36:00
Extracted first frame from 13:37:00
Extracted last frame from 13:37:00
Extracted first frame from 13:38:00
Extracted last frame from 13:38:00
Extracted first frame from 13:39:00
Extracted last frame from 13:39:00
Extracted first frame from 13:40:00
Extracted last frame from 13:40:00
Extracted first frame from 13:41:00
Extracted last frame from 13:41:00
Extracted first frame from 13:42:00
Extracted last frame from 13:42:00
Extracted first frame from 13:43:00
Extracted last frame from 13:43:00
Extracted first frame from 13:44:00
Extracted last frame from 13:44:00
```

Step 3: Image Feature Extraction

```
pip install torch torchvision openpyxl
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: torchvision in /usr/local/lib/python3.10/dist-p
Requirement already satisfied: openpyxl in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/pyth
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in /usr/local/lib/python3
Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/di
```

```
import torch
import torch.nn as nn
from torchvision import models, transforms
from PIL import Image
import pandas as pd
import os

# Load a pretrained ResNet-50 model
resnet_model = models.resnet50(pretrained=True)
resnet_model.eval()

feature_extractor = nn.Sequential(*list(resnet_model.children())[:-1])

transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

# Function to extract image features using ResNet-50
def extract_image_features(image_path):
    image = Image.open(image_path).convert("RGB")
    image_tensor = transform(image).unsqueeze(0)
    with torch.no_grad():
        features = feature_extractor(image_tensor)
    return features.squeeze().numpy()

def add_features_to_excel(image_folder, excel_path, output_excel_path):
    df = pd.read_excel(excel_path)

    feature_columns = [f"Feature_{i}" for i in range(2048)]

    feature_df = pd.DataFrame(columns=feature_columns)

    for idx, image_name in enumerate(os.listdir(image_folder)):
        image_path = os.path.join(image_folder, image_name)
        features = extract_image_features(image_path)
        feature_df.loc[idx] = features

    final_df = pd.concat([df, feature_df], axis=1)

    final_df.to_excel(output_excel_path, index=False)
    print(f"Features extracted and saved to {output_excel_path}")

image_folder = "/content/drive/MyDrive/Fire Detection/extracted_frames"
excel_path = "/content/drive/MyDrive/Fire Detection/cleaned_data.xlsx"
output_excel_path = "/content/drive/MyDrive/Fire Detection/output_with_features.x

add_features_to_excel(image_folder, excel_path, output_excel_path)
```

```
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:208: Use
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:223: Use
  warnings.warn(msg)
Features extracted and saved to /content/drive/MyDrive/Fire Detection/output_
```

```
import pandas as pd

# Load the Excel file
file_path = '/content/drive/MyDrive/Fire Detection/output_with_features.xlsx'
df = pd.read_excel(file_path)

# Step 1: Convert 'Unnamed: 0' to datetime and rename to 'Date'
if 'Unnamed: 0' in df.columns:
    df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'], errors='coerce')  # Conve
    df.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)

# Step 2: Drop any columns containing text data (non-numeric types other than dat
non_numeric_columns = df.select_dtypes(include=['object']).columns
df_cleaned = df.drop(columns=non_numeric_columns)

# Save the cleaned data to a new Excel file if needed
output_path = '/content/drive/MyDrive/Fire Detection/cleaned_data_with_features.x
df_cleaned.to_excel(output_path, index=False)

print(f"Cleaned data saved to {output_path}")
```

```
Cleaned data saved to /content/drive/MyDrive/Fire Detection/cleaned_data_with_
```

Step 4: Regression Model Development

For Implementation 1 : evaluate six to seven regression models and compare their performance based on these metrics, MSE and R²

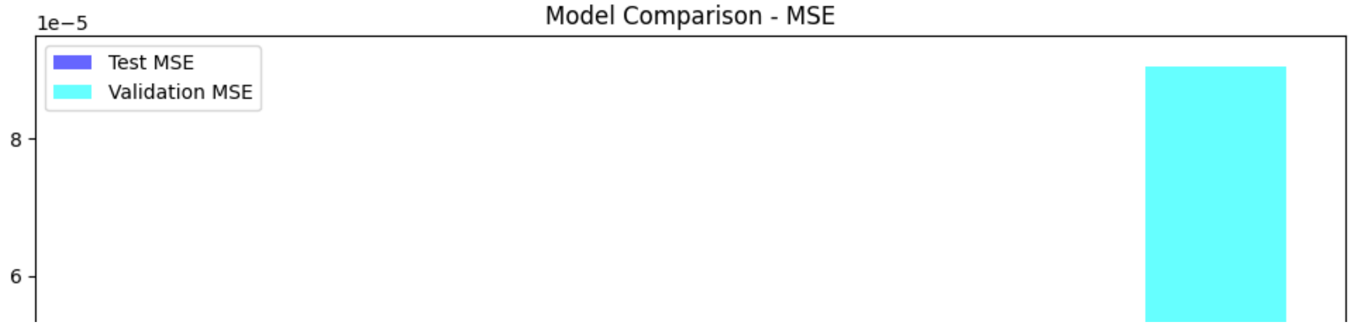
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
from google.colab import drive

# Mount Google Drive
```

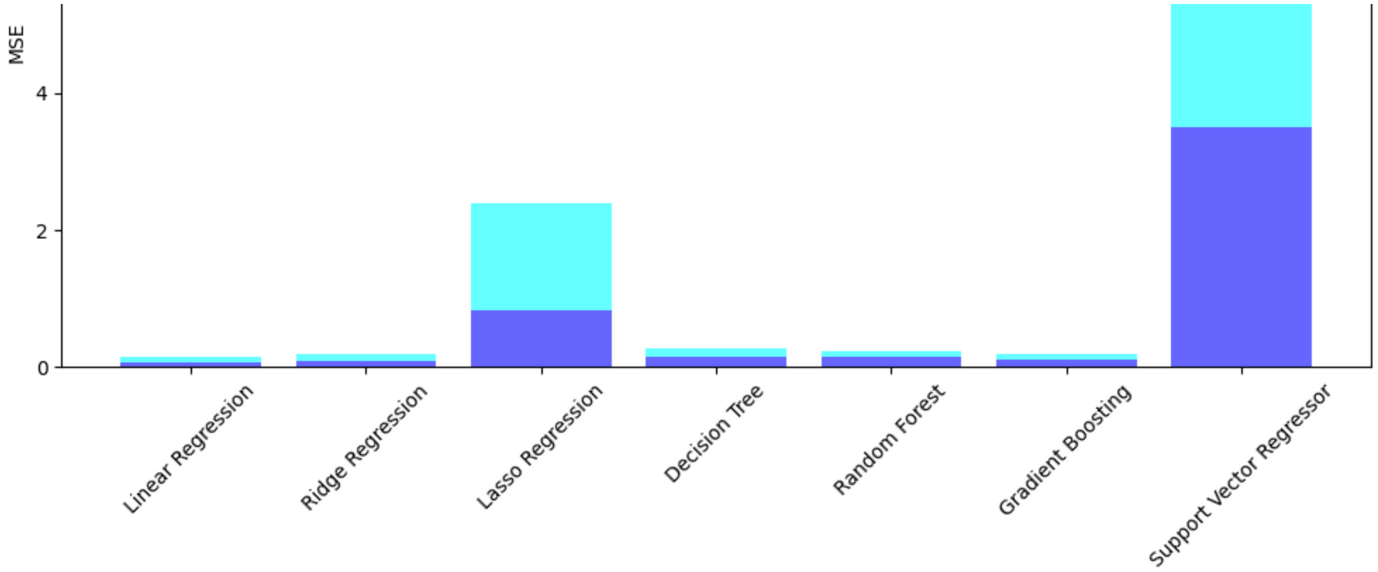
```
drive.mount('/content/drive')

# Load the Excel file
file_path = '/content/drive/MyDrive/Fire Detection/cleaned_data.xlsx'  # Using th
df = pd.read_excel(file_path)

# Data Preprocessing
if 'Unnamed: 0' in df.columns:
    df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'])
    df.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)

# Filter rows where 'CE' and 'DRE' are not zero
df_filtered = df[(df['CE'] != 0) & (df['DRE'] != 0)]

# Remove columns with constant values
constant_columns = [col for col in df_filtered.columns if df_filtered[col].nuniqu
df_filtered = df_filtered.drop(columns=constant_columns)

# Drop any non-numeric columns except 'Date'
non_numeric_columns = df_filtered.select_dtypes(include=['object']).columns
df_filtered = df_filtered.drop(columns=non_numeric_columns)

# Define features (X) and target (y) variables
X = df_filtered.drop(columns=['CE', 'DRE', 'Date'])
y_ce = df_filtered['CE']  # Target variable for CE
y_dre = df_filtered['DRE']  # Target variable for DRE

# Split data into training (80%), testing (10%), and validation (10%) sets
X_train, X_temp, y_ce_train, y_ce_temp = train_test_split(X, y_ce, test_size=0.2,
X_test, X_val, y_ce_test, y_ce_val = train_test_split(X_temp, y_ce_temp, test_siz

# Initialize models
models = {
    "Linear Regression": LinearRegression(),
    "Ridge Regression": Ridge(),
    "Lasso Regression": Lasso(),
    "Decision Tree": DecisionTreeRegressor(),
    "Random Forest": RandomForestRegressor(n_estimators=100, random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(),
    "Support Vector Regressor": SVR()
}

# Dictionary to store results
results = {}
```

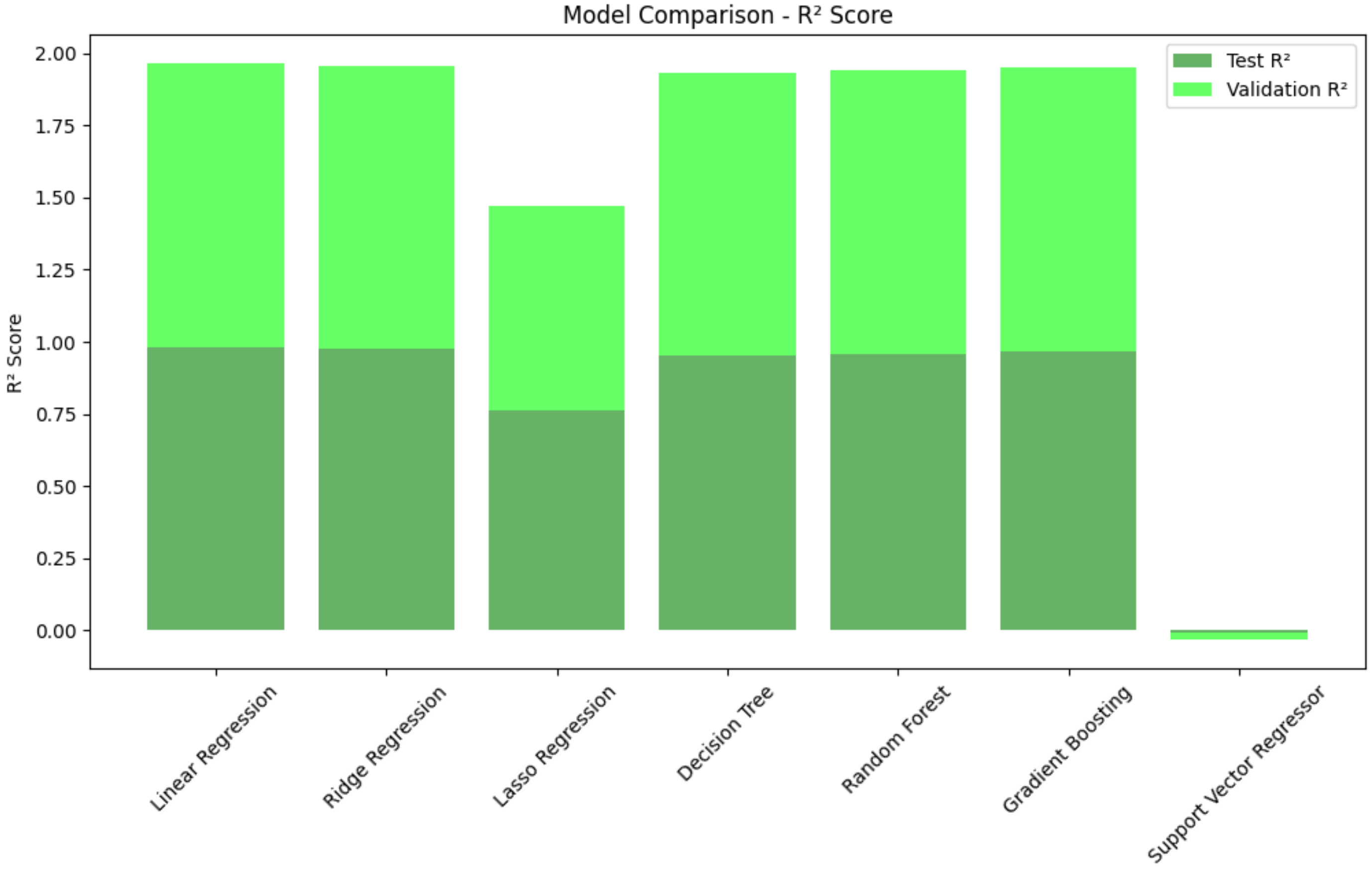
```
# Train and evaluate each model
for name, model in models.items():
    model.fit(X_train, y_ce_train)  # Train the model

    # Test set predictions and evaluations
    y_test_pred = model.predict(X_test)
    test_mse = mean_squared_error(y_ce_test, y_test_pred)
    test_r2 = r2_score(y_ce_test, y_test_pred)

    # Validation set predictions and evaluations
    y_val_pred = model.predict(X_val)
    val_mse = mean_squared_error(y_ce_val, y_val_pred)
    val_r2 = r2_score(y_ce_val, y_val_pred)

    # Store results
    results[name] = {
        "Test MSE": test_mse,
        "Test R2": test_r2,
        "Validation MSE": val_mse,
        "Validation R2": val_r2
    }

# Display results
for model_name, metrics in results.items():
    print(f"\n{model_name}")
    print(f"Test MSE: {metrics['Test MSE']:.4f}, Test R2: {metrics['Test R2']:.4f
    print(f"Validation MSE: {metrics['Validation MSE']:.4f}, Validation R2: {metr

# Visualization of MSE and R² for each model
model_names = list(results.keys())
test_mse_scores = [results[name]["Test MSE"] for name in model_names]
val_mse_scores = [results[name]["Validation MSE"] for name in model_names]
test_r2_scores = [results[name]["Test R2"] for name in model_names]
val_r2_scores = [results[name]["Validation R2"] for name in model_names]

# Plot MSE Comparison
plt.figure(figsize=(12, 6))
plt.bar(model_names, test_mse_scores, color='blue', alpha=0.6, label='Test MSE')
plt.bar(model_names, val_mse_scores, color='cyan', alpha=0.6, label='Validation M
plt.ylabel('MSE')
plt.title('Model Comparison - MSE')
plt.xticks(rotation=45)
plt.legend()
plt.show()
```

```
# Plot R² Comparison
plt.figure(figsize=(12, 6))
plt.bar(model_names, test_r2_scores, color='green', alpha=0.6, label='Test R²')
plt.bar(model_names, val_r2_scores, color='lime', alpha=0.6, label='Validation R²
plt.ylabel('R² Score')
plt.title('Model Comparison - R² Score')
plt.xticks(rotation=45)
plt.legend()
plt.show()
```

```
Mounted at /content/drive

Linear Regression
Test MSE: 0.0000, Test R2: 0.9799
Validation MSE: 0.0000, Validation R2: 0.9843

Ridge Regression
Test MSE: 0.0000, Test R2: 0.9748
Validation MSE: 0.0000, Validation R2: 0.9819

Lasso Regression
Test MSE: 0.0000, Test R2: 0.7621
Validation MSE: 0.0000, Validation R2: 0.7105

Decision Tree
Test MSE: 0.0000, Test R2: 0.9546
Validation MSE: 0.0000, Validation R2: 0.9799

Random Forest
Test MSE: 0.0000, Test R2: 0.9569
Validation MSE: 0.0000, Validation R2: 0.9865

Gradient Boosting
Test MSE: 0.0000, Test R2: 0.9689
Validation MSE: 0.0000, Validation R2: 0.9837

Support Vector Regressor
Test MSE: 0.0000, Test R2: -0.0100
Validation MSE: 0.0001, Validation R2: -0.0232
```




MSE
4
2
0
Linear Regression
Ridge Regression
Lasso Regression
Decision Tree
Random Forest
Gradient Boosting
Support Vector Regressor

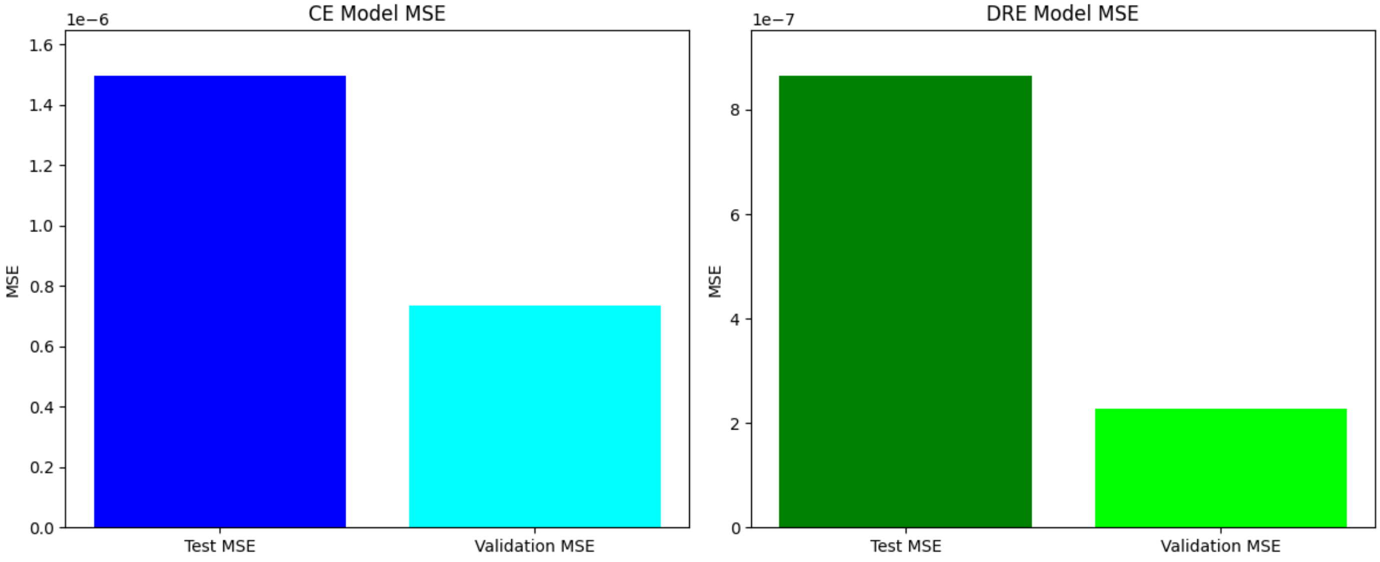

Model Comparison - R² Score
R² Score
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Test R²
Validation R²
Linear Regression
Ridge Regression
Lasso Regression
Decision Tree
Random Forest
Gradient Boosting
Support Vector Regressor

```
# implementation 1

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import numpy as np

# Load the Excel file
file_path = '/content/drive/MyDrive/Fire Detection/cleaned_data.xlsx'
df = pd.read_excel(file_path)

# Step 1: Ensure 'Unnamed: 0' (assuming it's the timestamp) is treated as a datet
if 'Unnamed: 0' in df.columns:
    df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'])
    df.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)

# Step 2: Filter rows where 'CE' and 'DRE' are not zero
df_filtered = df[(df['CE'] != 0) & (df['DRE'] != 0)]

# Step 3: Remove columns with constant values
constant_columns = [col for col in df_filtered.columns if df_filtered[col].nuniqu
df_filtered = df_filtered.drop(columns=constant_columns)

# Step 4: Drop any non-numeric columns except 'Date'
non_numeric_columns = df_filtered.select_dtypes(include=['object']).columns
df_filtered = df_filtered.drop(columns=non_numeric_columns)
```

```
# Step 5: Exclude 'Date' from X (independent variables)
X = df_filtered.drop(columns=['CE', 'DRE', 'Date'])
y_ce = df_filtered['CE']  # Target variable for CE
y_dre = df_filtered['DRE']  # Target variable for DRE

# Split the data into training (80%), testing (10%), and validation (10%) sets
X_train, X_temp, y_ce_train, y_ce_temp, y_dre_train, y_dre_temp = train_test_spli
X_test, X_val, y_ce_test, y_ce_val, y_dre_test, y_dre_val = train_test_split(X_te

# Initialize the regression model (using Random Forest as an example)
model_ce = RandomForestRegressor(n_estimators=100, random_state=42)
model_dre = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model for CE and DRE
model_ce.fit(X_train, y_ce_train)
model_dre.fit(X_train, y_dre_train)

# Predictions on test and validation sets
y_ce_test_pred = model_ce.predict(X_test)
y_ce_val_pred = model_ce.predict(X_val)
y_dre_test_pred = model_dre.predict(X_test)
y_dre_val_pred = model_dre.predict(X_val)

# Evaluate the models
test_mse_ce = mean_squared_error(y_ce_test, y_ce_test_pred)
val_mse_ce = mean_squared_error(y_ce_val, y_ce_val_pred)
test_r2_ce = r2_score(y_ce_test, y_ce_test_pred)
val_r2_ce = r2_score(y_ce_val, y_ce_val_pred)

test_mse_dre = mean_squared_error(y_dre_test, y_dre_test_pred)
val_mse_dre = mean_squared_error(y_dre_val, y_dre_val_pred)
test_r2_dre = r2_score(y_dre_test, y_dre_test_pred)
val_r2_dre = r2_score(y_dre_val, y_dre_val_pred)

# Print model evaluation results
print("CE Model Evaluation:")
print(f"Test MSE: {test_mse_ce}, Test R2: {test_r2_ce}")
print(f"Validation MSE: {val_mse_ce}, Validation R2: {val_r2_ce}")

print("\nDRE Model Evaluation:")
print(f"Test MSE: {test_mse_dre}, Test R2: {test_r2_dre}")
print(f"Validation MSE: {val_mse_dre}, Validation R2: {val_r2_dre}")

# Plotting MSE for CE and DRE on test and validation sets
```

```
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Plot CE MSE
axes[0].bar(['Test MSE', 'Validation MSE'], [test_mse_ce, val_mse_ce], color=['bl
axes[0].set_title("CE Model MSE")
axes[0].set_ylabel("MSE")
axes[0].set_ylim(0, max(test_mse_ce, val_mse_ce) * 1.1)

# Plot DRE MSE
axes[1].bar(['Test MSE', 'Validation MSE'], [test_mse_dre, val_mse_dre], color=['
axes[1].set_title("DRE Model MSE")
axes[1].set_ylabel("MSE")
axes[1].set_ylim(0, max(test_mse_dre, val_mse_dre) * 1.1)

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

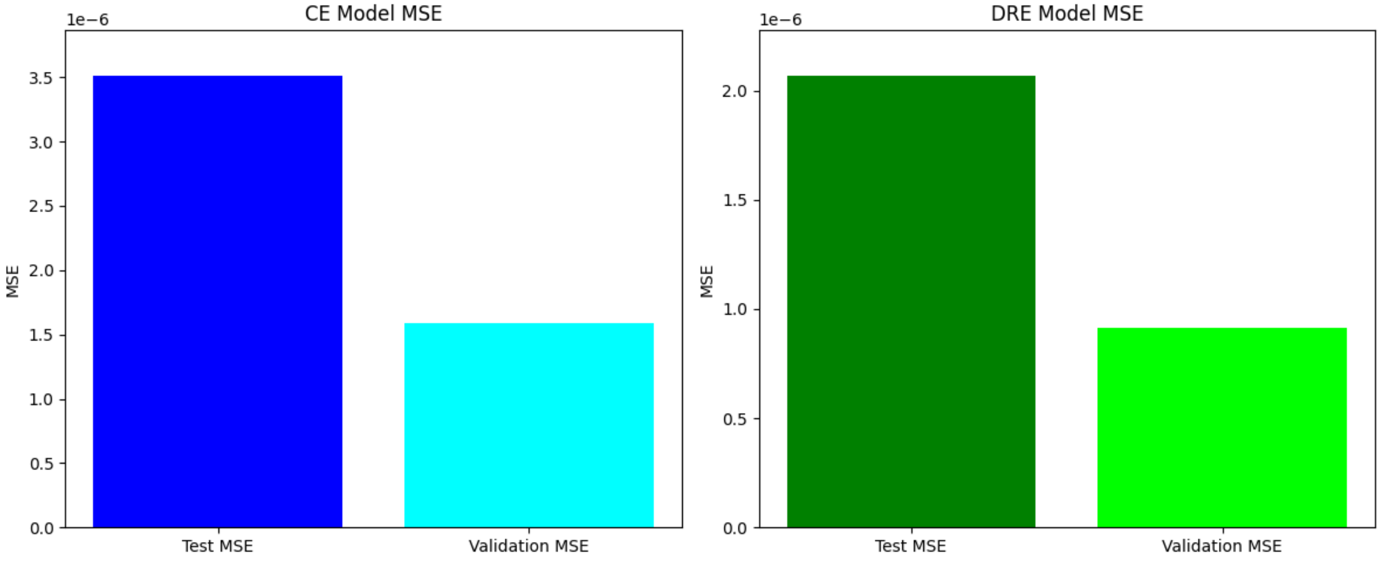
```
CE Model Evaluation:
Test MSE: 1.4967989265287115e-06, Test R2: 0.9568870409675998
Validation MSE: 7.336629923547264e-07, Validation R2: 0.9864639256043718

DRE Model Evaluation:
Test MSE: 8.653649052746361e-07, Test R2: 0.9658111081975436
Validation MSE: 2.2702495661856075e-07, Validation R2: 0.9931387492789988
```



```
# implementation 2

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load the Excel file containing both Excel and image features
file_path = '/content/drive/MyDrive/Fire Detection/cleaned_data_with_features.xls
df = pd.read_excel(file_path)

# Step 1: Convert 'Unnamed: 0' to datetime and rename it to 'Date' for consistenc
if 'Unnamed: 0' in df.columns:
    df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'], errors='coerce')
    df.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)

# Step 2: Filter Data Range (13:15 to 16:15)
start_time = '2022-10-05 13:15:00'
end_time = '2022-10-05 16:15:00'
df_filtered = df[(df['Date'] >= start_time) & (df['Date'] <= end_time)]

# Step 3: Drop columns with text data, keeping only numeric columns and the 'Date
non_numeric_columns = df_filtered.select_dtypes(include=['object']).columns
df_filtered = df_filtered.drop(columns=non_numeric_columns)

# Step 4: Define independent (X) and dependent (y) variables
# Select columns starting with 'Feature_' and other relevant columns (excluding '
feature_columns = [col for col in df_filtered.columns if col.startswith('Feature_
other_columns = [col for col in df_filtered.columns if col not in ['CE', 'DRE', '
X = df_filtered[feature_columns + other_columns]
y_ce = df_filtered['CE']  # Target for CE
y_dre = df_filtered['DRE']  # Target for DRE

# Step 5: Train-Test-Validation Split (80% training, 10% testing, 10% validation)
X_train, X_temp, y_ce_train, y_ce_temp, y_dre_train, y_dre_temp = train_test_spli
X_test, X_val, y_ce_test, y_ce_val, y_dre_test, y_dre_val = train_test_split(X_te

# Step 6: Initialize the regression model (using Random Forest for CE and DRE pre
model_ce = RandomForestRegressor(n_estimators=100, random_state=42)
model_dre = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model for CE and DRE
model_ce.fit(X_train, y_ce_train)
model_dre.fit(X_train, y_dre_train)

# Step 7: Predictions on test and validation sets
y_ce_test_pred = model_ce.predict(X_test)
y_ce_val_pred = model_ce.predict(X_val)
y_dre_test_pred = model_dre.predict(X_test)
y_dre_val_pred = model_dre.predict(X_val)

# Step 8: Evaluate the models for CE
test_mse_ce = mean_squared_error(y_ce_test, y_ce_test_pred)
val_mse_ce = mean_squared_error(y_ce_val, y_ce_val_pred)
test_r2_ce = r2_score(y_ce_test, y_ce_test_pred)
val_r2_ce = r2_score(y_ce_val, y_ce_val_pred)

# Evaluate the models for DRE
```

```
test_mse_dre = mean_squared_error(y_dre_test, y_dre_test_pred)
val_mse_dre = mean_squared_error(y_dre_val, y_dre_val_pred)
test_r2_dre = r2_score(y_dre_test, y_dre_test_pred)
val_r2_dre = r2_score(y_dre_val, y_dre_val_pred)

# Print model evaluation results
print("CE Model Evaluation:")
print(f"Test MSE: {test_mse_ce}, Test R2: {test_r2_ce}")
print(f"Validation MSE: {val_mse_ce}, Validation R2: {val_r2_ce}")

print("\nDRE Model Evaluation:")
print(f"Test MSE: {test_mse_dre}, Test R2: {test_r2_dre}")
print(f"Validation MSE: {val_mse_dre}, Validation R2: {val_r2_dre}")

# Step 9: Plotting MSE for CE and DRE on test and validation sets
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Plot CE MSE
axes[0].bar(['Test MSE', 'Validation MSE'], [test_mse_ce, val_mse_ce], color=['bl
axes[0].set_title("CE Model MSE")
axes[0].set_ylabel("MSE")
axes[0].set_ylim(0, max(test_mse_ce, val_mse_ce) * 1.1)

# Plot DRE MSE
axes[1].bar(['Test MSE', 'Validation MSE'], [test_mse_dre, val_mse_dre], color=['
axes[1].set_title("DRE Model MSE")
axes[1].set_ylabel("MSE")
axes[1].set_ylim(0, max(test_mse_dre, val_mse_dre) * 1.1)

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

```
CE Model Evaluation:
Test MSE: 3.5121638179741306e-06, Test R2: 0.8988375979460643
Validation MSE: 1.5899951475905114e-06, Validation R2: 0.970664606459982

DRE Model Evaluation:
Test MSE: 2.0688664550309694e-06, Test R2: 0.9182630923051618
Validation MSE: 9.156109873657216e-07, Validation R2: 0.9723279914208952
```



```
image_dir = '/content/drive/MyDrive/Fire Detection/extracted_frames'
```

```
import pandas as pd
import numpy as np
import cv2
import os
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
import joblib
```

```
# Define the directory containing images
image_dir = '/content/drive/MyDrive/Fire Detection/new_extracted_frames'

# Extract KPI features from each image
def extract_kpis_from_image(image_path):
    image = cv2.imread(image_path)
    if image is None:
        return None
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    contours, _ = cv2.findContours(gray, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMP
    angle = cv2.minAreaRect(contours[0])[-1] if contours else 0
    ratio = (cv2.boundingRect(contours[0])[2] / cv2.boundingRect(contours[0])[3])
    orientation = (cv2.phase(np.array([cv2.moments(contours[0])["m10"]]), np.arra
    color = image.mean(axis=0).mean(axis=0)[0]
    hist = cv2.calcHist([gray], [0], None, [256], [0, 256]).flatten()
    return {'angle': angle, 'ratio': ratio, 'orientation': orientation, 'color':

# Initialize list to hold KPI data
kpi_data = []
base_date = datetime.strptime("2022-10-05", "%Y-%m-%d")

# Process each image file in the directory
for filename in os.listdir(image_dir):
    if filename.endswith(".jpg"):
        print(f"Processing file: {filename}")
        try:
            time_str = filename.split('_')[1]
            timestamp = datetime.strptime(time_str, "%H:%M:%S")
            full_timestamp = base_date + timedelta(hours=timestamp.hour, minutes=

            image_path = os.path.join(image_dir, filename)
            kpis = extract_kpis_from_image(image_path)
            if kpis is not None:
                kpis['Date'] = full_timestamp
                kpi_data.append(kpis)
        except ValueError as e:
            print(f"Skipping file {filename} due to error: {e}")

# Create a DataFrame from the KPI data
kpi_df = pd.DataFrame(kpi_data)

# Load the Excel file containing numerical features
file_path = '/content/drive/MyDrive/Fire Detection/cleaned_data_with_features.xls
df = pd.read_excel(file_path)
```

```
# Convert 'Unnamed: 0' to datetime and rename it to 'Date'
if 'Unnamed: 0' in df.columns:
    df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'], errors='coerce')
    df.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)

# Filter rows within the required date range
start_time = '2022-10-05 13:15:00'
end_time = '2022-10-05 16:15:00'
df_filtered = df[(df['Date'] >= start_time) & (df['Date'] <= end_time)]
non_numeric_columns = df_filtered.select_dtypes(include=['object']).columns
df_filtered = df_filtered.drop(columns=non_numeric_columns)

# Merge KPI data with the filtered data
if 'Date' in df_filtered.columns and 'Date' in kpi_df.columns:
    df_combined = pd.merge(df_filtered, kpi_df, on='Date', how='inner')
else:
    print("Error: 'Date' column missing from one of the DataFrames.")
    raise KeyError("The 'Date' column must be present in both df_filtered and kpi

# Define features and target variables
feature_columns = [col for col in df_combined.columns if col.startswith('Feature_
kpi_columns = ['angle', 'ratio', 'orientation', 'color', 'histogram']
other_columns = [col for col in df_combined.columns if col not in ['CE', 'DRE', '
X = df_combined[feature_columns + other_columns + kpi_columns]
y_ce = df_combined['CE']
y_dre = df_combined['DRE']

# Train-Test-Validation Split
X_train, X_temp, y_ce_train, y_ce_temp, y_dre_train, y_dre_temp = train_test_spli
X_test, X_val, y_ce_test, y_ce_val, y_dre_test, y_dre_val = train_test_split(X_te

# Initialize regression models
model_ce = RandomForestRegressor(n_estimators=100, random_state=42)
model_dre = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the models
model_ce.fit(X_train, y_ce_train)
model_dre.fit(X_train, y_dre_train)

# Make predictions and calculate metrics
y_ce_test_pred = model_ce.predict(X_test)
y_ce_val_pred = model_ce.predict(X_val)
y_dre_test_pred = model_dre.predict(X_test)
y_dre_val_pred = model_dre.predict(X_val)
```

```
# Evaluation results
results = {
    "CE Test MSE": mean_squared_error(y_ce_test, y_ce_test_pred),
    "CE Test R2": r2_score(y_ce_test, y_ce_test_pred),
    "CE Validation MSE": mean_squared_error(y_ce_val, y_ce_val_pred),
    "CE Validation R2": r2_score(y_ce_val, y_ce_val_pred),
    "DRE Test MSE": mean_squared_error(y_dre_test, y_dre_test_pred),
    "DRE Test R2": r2_score(y_dre_test, y_dre_test_pred),
    "DRE Validation MSE": mean_squared_error(y_dre_val, y_dre_val_pred),
    "DRE Validation R2": r2_score(y_dre_val, y_dre_val_pred),
}

# Save the results to a new Excel file
results_df = pd.DataFrame([results])
results_file_path = '/content/drive/MyDrive/Fire Detection/model_evaluation_resul
results_df.to_excel(results_file_path, index=False)
print(f"Results saved to {results_file_path}")

# Save the trained models
joblib.dump(model_ce, '/content/drive/MyDrive/Fire Detection/model_ce.joblib')
joblib.dump(model_dre, '/content/drive/MyDrive/Fire Detection/model_dre.joblib')
print("Models saved successfully.")

# Plot results
fig, axes = plt.subplots(1, 2, figsize=(12, 5))
axes[0].bar(['Test MSE', 'Validation MSE'], [results["CE Test MSE"], results["CE
axes[0].set_title("CE Model MSE with KPIs")
axes[0].set_ylabel("MSE")
axes[1].bar(['Test MSE', 'Validation MSE'], [results["DRE Test MSE"], results["DR
axes[1].set_title("DRE Model MSE with KPIs")
axes[1].set_ylabel("MSE")
plt.tight_layout()
plt.show()
```

```
Processing file: frame_13:15:00_first.jpg
Processing file: frame_13:15:00_last.jpg
Processing file: frame_13:16:00_first.jpg
Processing file: frame_13:16:00_last.jpg
Processing file: frame_13:17:00_first.jpg
Processing file: frame_13:17:00_last.jpg
Processing file: frame_13:18:00_first.jpg
Processing file: frame_13:18:00_last.jpg
Processing file: frame_13:19:00_first.jpg
Processing file: frame_13:19:00_last.jpg
Processing file: frame_13:20:00_first.jpg
Processing file: frame_13:20:00_last.jpg
Processing file: frame_13:21:00_first.jpg
```

```
        Processing file: frame_13:21:00_first.jpg
        Processing file: frame_13:21:00_last.jpg
        Processing file: frame_13:22:00_first.jpg
        Processing file: frame_13:22:00_last.jpg
        Processing file: frame_13:23:00_first.jpg
        Processing file: frame_13:23:00_last.jpg
        Processing file: frame_13:24:00_first.jpg
        Processing file: frame_13:24:00_last.jpg
        Processing file: frame_13:25:00_first.jpg
        Processing file: frame_13:25:00_last.jpg
        Processing file: frame_13:26:00_first.jpg
        Processing file: frame_13:26:00_last.jpg
        Processing file: frame_13:27:00_first.jpg
        Processing file: frame_13:27:00_last.jpg
        Processing file: frame_13:28:00_first.jpg
        Processing file: frame_13:28:00_last.jpg
        Processing file: frame_13:29:00_first.jpg
        Processing file: frame_13:29:00_last.jpg
        Processing file: frame_13:30:00_first.jpg
        Processing file: frame_13:30:00_last.jpg
        Processing file: frame_13:31:00_first.jpg
        Processing file: frame_13:31:00_last.jpg
        Processing file: frame_13:32:00_first.jpg
        Processing file: frame_13:32:00_last.jpg
        Processing file: frame_13:33:00_first.jpg
        Processing file: frame_13:33:00_last.jpg
        Processing file: frame_13:34:00_first.jpg
        Processing file: frame_13:34:00_last.jpg
        Processing file: frame_13:35:00_first.jpg
        Processing file: frame_13:35:00_last.jpg
        Processing file: frame_13:36:00_first.jpg
        Processing file: frame_13:36:00_last.jpg
        Processing file: frame_13:37:00_first.jpg
        Processing file: frame_13:37:00_last.jpg
        Processing file: frame_13:38:00_first.jpg
        Processing file: frame_13:38:00_last.jpg
        Processing file: frame_13:39:00_first.jpg
        Processing file: frame_13:39:00_last.jpg
        Processing file: frame_13:40:00_first.jpg
        Processing file: frame_13:40:00_last.jpg
        Processing file: frame_13:41:00_first.jpg
        Processing file: frame_13:41:00_last.jpg
        Processing file: frame_13:42:00_first.jpg
        Processing file: frame_13:42:00_last.jpg
        Processing file: frame_13:43:00_first.jpg
        Processing file: frame_13:43:00_last.jpg
        Processing file: frame_13:44:00_first.jpg
        Processing file: frame_13:44:00_last.jpg
        Processing file: frame_13:45:00_first.jpg
        Processing file: frame_13:45:00_last.jpg
        Processing file: frame_13:46:00_first.jpg
        Processing file: frame_13:46:00_last.jpg
        Processing file: frame_13:47:00_first.jpg
        Processing file: frame_13:47:00_last.jpg
        Processing file: frame_13:48:00_first.jpg
        Processing file: frame_13:48:00_last.jpg
        Processing file: frame_13:49:00_first.jpg
        Processing file: frame_13:49:00_last.jpg
        Processing file: frame_13:50:00_first.jpg
        Processing file: frame_13:50:00_last.jpg
        Processing file: frame_13:51:00_first.jpg
        Processing file: frame_13:51:00_last.jpg
        Processing file: frame_13:52:00_first.jpg
        Processing file: frame_13:52:00_last.jpg
        Processing file: frame_13:53:00_first.jpg
        Processing file: frame_13:53:00_last.jpg
        Processing file: frame_13:54:00_first.jpg
        Processing file: frame_13:54:00_last.jpg
        Processing file: frame_13:55:00_first.jpg
        Processing file: frame_13:55:00_last.jpg
        Processing file: frame_13:56:00_first.jpg
        Processing file: frame_13:56:00_last.jpg
        Processing file: frame_13:57:00_first.jpg
        Processing file: frame_13:57:00_last.jpg
        Processing file: frame_13:58:00_first.jpg
        Processing file: frame_13:58:00_last.jpg
        Processing file: frame_13:59:00_first.jpg
        Processing file: frame_13:59:00_last.jpg
        Processing file: frame_14:00:00_first.jpg
        Processing file: frame_14:00:00_last.jpg
        Processing file: frame_14:01:00_first.jpg
        Processing file: frame_14:01:00_last.jpg
        Processing file: frame_14:02:00_first.jpg
        Processing file: frame_14:02:00_last.jpg
        Processing file: frame_14:03:00_first.jpg
        Processing file: frame_14:03:00_last.jpg
        Processing file: frame_14:04:00_first.jpg
        Processing file: frame_14:04:00_last.jpg
        Processing file: frame_14:05:00_first.jpg
        Processing file: frame_14:05:00_last.jpg
        Processing file: frame_14:06:00_first.jpg
        Processing file: frame_14:06:00_last.jpg
        Processing file: frame_14:07:00_first.jpg
        Processing file: frame_14:07:00_last.jpg
        Processing file: frame_14:08:00_first.jpg
        Processing file: frame_14:08:00_last.jpg
        Processing file: frame_14:09:00_first.jpg
        Processing file: frame_14:09:00_last.jpg
        Processing file: frame_14:10:00_first.jpg
        Processing file: frame_14:10:00_last.jpg
        Processing file: frame_14:11:00_first.jpg
```

```
Processing file: frame_14:11:00_last.jpg
Processing file: frame_14:12:00_first.jpg
Processing file: frame_14:12:00_last.jpg
Processing file: frame_14:13:00_first.jpg
Processing file: frame_14:13:00_last.jpg
Processing file: frame_14:14:00_first.jpg
Processing file: frame_14:14:00_last.jpg
Processing file: frame_14:15:00_first.jpg
Processing file: frame_14:15:00_last.jpg
Processing file: frame_14:16:00_first.jpg
Processing file: frame_14:16:00_last.jpg
Processing file: frame_14:17:00_first.jpg
Processing file: frame_14:17:00_last.jpg
Processing file: frame_14:18:00_first.jpg
Processing file: frame_14:18:00_last.jpg
Processing file: frame_14:19:00_first.jpg
Processing file: frame_14:19:00_last.jpg
Processing file: frame_14:20:00_first.jpg
Processing file: frame_14:20:00_last.jpg
Processing file: frame_14:21:00_first.jpg
Processing file: frame_14:21:00_last.jpg
Processing file: frame_14:22:00_first.jpg
Processing file: frame_14:22:00_last.jpg
Processing file: frame_14:23:00_first.jpg
Processing file: frame_14:23:00_last.jpg
Processing file: frame_14:24:00_first.jpg
Processing file: frame_14:24:00_last.jpg
Processing file: frame_14:25:00_first.jpg
Processing file: frame_14:25:00_last.jpg
Processing file: frame_14:26:00_first.jpg
Processing file: frame_14:26:00_last.jpg
Processing file: frame_14:27:00_first.jpg
Processing file: frame_14:27:00_last.jpg
Processing file: frame_14:28:00_first.jpg
Processing file: frame_14:28:00_last.jpg
Processing file: frame_14:29:00_first.jpg
Processing file: frame_14:29:00_last.jpg
Processing file: frame_14:30:00_first.jpg
Processing file: frame_14:30:00_last.jpg
Processing file: frame_14:31:00_first.jpg
Processing file: frame_14:31:00_last.jpg
Processing file: frame_14:32:00_first.jpg
Processing file: frame_14:32:00_last.jpg
Processing file: frame_14:33:00_first.jpg
Processing file: frame_14:33:00_last.jpg
Processing file: frame_14:34:00_first.jpg
Processing file: frame_14:34:00_last.jpg
Processing file: frame_14:35:00_first.jpg
Processing file: frame_14:35:00_last.jpg
Processing file: frame_14:36:00_first.jpg
Processing file: frame_14:36:00_last.jpg
Processing file: frame_14:36:00_last.jpg
Processing file: frame_14:37:00_first.jpg
Processing file: frame_14:37:00_last.jpg
Processing file: frame_14:38:00_first.jpg
Processing file: frame_14:38:00_last.jpg
Processing file: frame_14:39:00_first.jpg
Processing file: frame_14:39:00_last.jpg
Processing file: frame_14:40:00_first.jpg
Processing file: frame_14:40:00_last.jpg
Processing file: frame_14:41:00_first.jpg
Processing file: frame_14:41:00_last.jpg
Processing file: frame_14:42:00_first.jpg
Processing file: frame_14:42:00_last.jpg
Processing file: frame_14:43:00_first.jpg
Processing file: frame_14:43:00_last.jpg
Processing file: frame_14:44:00_first.jpg
Processing file: frame_14:44:00_last.jpg
Processing file: frame_14:45:00_first.jpg
Processing file: frame_14:45:00_last.jpg
Processing file: frame_14:46:00_first.jpg
Processing file: frame_14:46:00_last.jpg
Processing file: frame_14:47:00_first.jpg
Processing file: frame_14:47:00_last.jpg
Processing file: frame_14:48:00_first.jpg
Processing file: frame_14:48:00_last.jpg
Processing file: frame_14:49:00_first.jpg
Processing file: frame_14:49:00_last.jpg
Processing file: frame_14:50:00_first.jpg
Processing file: frame_14:50:00_last.jpg
Processing file: frame_14:51:00_first.jpg
Processing file: frame_14:51:00_last.jpg
Processing file: frame_14:52:00_first.jpg
Processing file: frame_14:52:00_last.jpg
Processing file: frame_14:53:00_first.jpg
Processing file: frame_14:53:00_last.jpg
Processing file: frame_14:54:00_first.jpg
Processing file: frame_14:54:00_last.jpg
Processing file: frame_14:55:00_first.jpg
Processing file: frame_14:55:00_last.jpg
Processing file: frame_14:56:00_first.jpg
Processing file: frame_14:56:00_last.jpg
Processing file: frame_14:57:00_first.jpg
Processing file: frame_14:57:00_last.jpg
Processing file: frame_14:58:00_first.jpg
Processing file: frame_14:58:00_last.jpg
Processing file: frame_14:59:00_first.jpg
Processing file: frame_14:59:00_last.jpg
Processing file: frame_15:00:00_first.jpg
Processing file: frame_15:00:00_last.jpg
Processing file: frame_15:01:00_first.jpg
Processing file: frame_15:01:00_last.jpg
Processing file: frame_15:02:00_first.jpg
Processing file: frame_15:02:00_last.jpg
Processing file: frame_15:03:00_first.jpg
Processing file: frame_15:03:00_last.jpg
Processing file: frame_15:04:00_first.jpg
Processing file: frame_15:04:00_last.jpg
Processing file: frame_15:05:00_first.jpg
Processing file: frame_15:05:00_last.jpg
Processing file: frame_15:06:00_first.jpg
Processing file: frame_15:06:00_last.jpg
Processing file: frame_15:07:00_first.jpg
Processing file: frame_15:07:00_last.jpg
Processing file: frame_15:08:00_first.jpg
Processing file: frame_15:08:00_last.jpg
Processing file: frame_15:09:00_first.jpg
Processing file: frame_15:09:00_last.jpg
Processing file: frame_15:10:00_first.jpg
Processing file: frame_15:10:00_last.jpg
Processing file: frame_15:11:00_first.jpg
Processing file: frame_15:11:00_last.jpg
Processing file: frame_15:12:00_first.jpg
Processing file: frame_15:12:00_last.jpg
Processing file: frame_15:13:00_first.jpg
Processing file: frame_15:13:00_last.jpg
Processing file: frame_15:14:00_first.jpg
Processing file: frame_15:14:00_last.jpg
Processing file: frame_15:15:00_first.jpg
Processing file: frame_15:15:00_last.jpg
Processing file: frame_15:16:00_first.jpg
Processing file: frame_15:16:00_last.jpg
Processing file: frame_15:17:00_first.jpg
Processing file: frame_15:17:00_last.jpg
Processing file: frame_15:18:00_first.jpg
Processing file: frame_15:18:00_last.jpg
Processing file: frame_15:19:00_first.jpg
Processing file: frame_15:19:00_last.jpg
Processing file: frame_15:20:00_first.jpg
Processing file: frame_15:20:00_last.jpg
Processing file: frame_15:21:00_first.jpg
Processing file: frame_15:21:00_last.jpg
Processing file: frame_15:22:00_first.jpg
Processing file: frame_15:22:00_last.jpg
Processing file: frame_15:23:00_first.jpg
Processing file: frame_15:23:00_last.jpg
Processing file: frame_15:24:00_first.jpg
Processing file: frame_15:24:00_last.jpg
Processing file: frame_15:25:00_first.jpg
Processing file: frame_15:25:00_last.jpg
Processing file: frame_15:26:00_first.jpg
Processing file: frame_15:26:00_last.jpg
Processing file: frame_15:27:00_first.jpg
```

```
        Processing file: frame_15:27:00_first.jpg
        Processing file: frame_15:27:00_last.jpg
        Processing file: frame_15:28:00_first.jpg
        Processing file: frame_15:28:00_last.jpg
        Processing file: frame_15:29:00_first.jpg
        Processing file: frame_15:29:00_last.jpg
        Processing file: frame_15:30:00_first.jpg
        Processing file: frame_15:30:00_last.jpg
        Processing file: frame_15:31:00_first.jpg
        Processing file: frame_15:31:00_last.jpg
        Processing file: frame_15:32:00_first.jpg
        Processing file: frame_15:32:00_last.jpg
        Processing file: frame_15:33:00_first.jpg
        Processing file: frame_15:33:00_last.jpg
        Processing file: frame_15:34:00_first.jpg
        Processing file: frame_15:34:00_last.jpg
        Processing file: frame_15:35:00_first.jpg
        Processing file: frame_15:35:00_last.jpg
        Processing file: frame_15:36:00_first.jpg
        Processing file: frame_15:36:00_last.jpg
        Processing file: frame_15:37:00_first.jpg
        Processing file: frame_15:37:00_last.jpg
        Processing file: frame_15:38:00_first.jpg
        Processing file: frame_15:38:00_last.jpg
        Processing file: frame_15:39:00_first.jpg
        Processing file: frame_15:39:00_last.jpg
        Processing file: frame_15:40:00_first.jpg
        Processing file: frame_15:40:00_last.jpg
        Processing file: frame_15:41:00_first.jpg
        Processing file: frame_15:41:00_last.jpg
        Processing file: frame_15:42:00_first.jpg
        Processing file: frame_15:42:00_last.jpg
        Processing file: frame_15:43:00_first.jpg
        Processing file: frame_15:43:00_last.jpg
        Processing file: frame_15:44:00_first.jpg
        Processing file: frame_15:44:00_last.jpg
        Processing file: frame_15:45:00_first.jpg
        Processing file: frame_15:45:00_last.jpg
        Processing file: frame_15:46:00_first.jpg
        Processing file: frame_15:46:00_last.jpg
        Processing file: frame_15:47:00_first.jpg
        Processing file: frame_15:47:00_last.jpg
        Processing file: frame_15:48:00_first.jpg
        Processing file: frame_15:48:00_last.jpg
        Processing file: frame_15:49:00_first.jpg
        Processing file: frame_15:49:00_last.jpg
        Processing file: frame_15:50:00_first.jpg
        Processing file: frame_15:50:00_last.jpg
        Processing file: frame_15:51:00_first.jpg
        Processing file: frame_15:51:00_last.jpg
        Processing file: frame 15:52:00 first.jpg
```

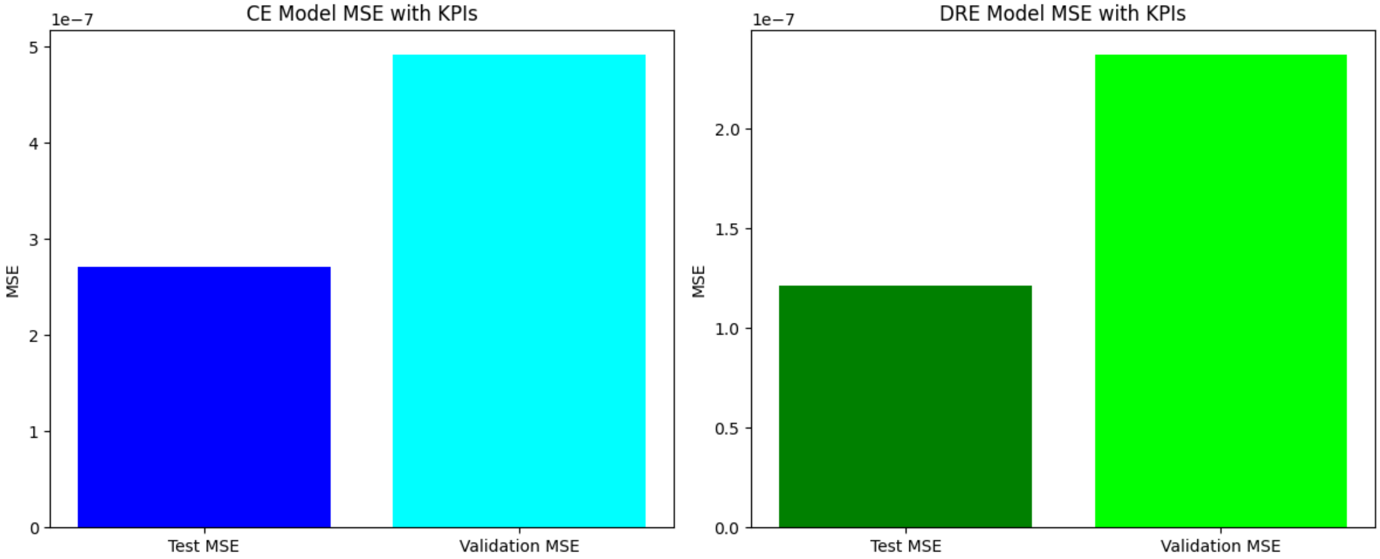
```
Processing file: frame_15:52:00_last.jpg
Processing file: frame_15:53:00_first.jpg
Processing file: frame_15:53:00_last.jpg
Processing file: frame_15:54:00_first.jpg
Processing file: frame_15:54:00_last.jpg
Processing file: frame_15:55:00_first.jpg
Processing file: frame_15:55:00_last.jpg
Processing file: frame_15:56:00_first.jpg
Processing file: frame_15:56:00_last.jpg
Processing file: frame_15:57:00_first.jpg
Processing file: frame_15:57:00_last.jpg
Processing file: frame_15:58:00_first.jpg
Processing file: frame_15:58:00_last.jpg
Processing file: frame_15:59:00_first.jpg
Processing file: frame_15:59:00_last.jpg
Processing file: frame_16:00:00_first.jpg
Processing file: frame_16:00:00_last.jpg
Processing file: frame_16:01:00_first.jpg
Processing file: frame_16:01:00_last.jpg
Processing file: frame_16:02:00_first.jpg
Processing file: frame_16:02:00_last.jpg
Processing file: frame_16:03:00_first.jpg
Processing file: frame_16:03:00_last.jpg
Processing file: frame_16:04:00_first.jpg
Processing file: frame_16:04:00_last.jpg
Processing file: frame_16:05:00_first.jpg
Processing file: frame_16:05:00_last.jpg
Processing file: frame_16:06:00_first.jpg
Processing file: frame_16:06:00_last.jpg
Processing file: frame_16:07:00_first.jpg
Processing file: frame_16:07:00_last.jpg
Processing file: frame_16:08:00_first.jpg
Processing file: frame_16:08:00_last.jpg
Processing file: frame_16:09:00_first.jpg
Processing file: frame_16:09:00_last.jpg
Processing file: frame_16:10:00_first.jpg
Processing file: frame_16:10:00_last.jpg
Processing file: frame_16:11:00_first.jpg
Processing file: frame_16:11:00_last.jpg
Processing file: frame_16:12:00_first.jpg
Processing file: frame_16:12:00_last.jpg
Processing file: frame_16:13:00_first.jpg
Processing file: frame_16:13:00_last.jpg
Processing file: frame_16:14:00_first.jpg
Processing file: frame_16:14:00_last.jpg
Processing file: frame_16:15:00_first.jpg
Processing file: frame_16:15:00_last.jpg
Processing file: frame_16:16:00_first.jpg
Processing file: frame_16:16:00_last.jpg
Processing file: frame_16:17:00_first.jpg
Processing file: frame_16:17:00_last.jpg
```

```
    Processing file: frame_16:17:00_last.jpg
    Processing file: frame_16:18:00_first.jpg
    Processing file: frame_16:18:00_last.jpg
    Processing file: frame_16:19:00_first.jpg
    Processing file: frame_16:19:00_last.jpg
    Processing file: frame_16:20:00_first.jpg
    Processing file: frame_16:20:00_last.jpg
    Processing file: frame_16:21:00_first.jpg
    Processing file: frame_16:21:00_last.jpg
    Processing file: frame_16:22:00_first.jpg
    Processing file: frame_16:22:00_last.jpg
    Processing file: frame_16:23:00_first.jpg
    Processing file: frame_16:23:00_last.jpg
    Processing file: frame_16:24:00_first.jpg
    Processing file: frame_16:24:00_last.jpg
    Processing file: frame_16:25:00_first.jpg
    Processing file: frame_16:25:00_last.jpg
    Processing file: frame_16:26:00_first.jpg
    Processing file: frame_16:26:00_last.jpg
    Processing file: frame_16:27:00_first.jpg
    Processing file: frame_16:27:00_last.jpg
    Processing file: frame_16:28:00_first.jpg
    Processing file: frame_16:28:00_last.jpg
    Processing file: frame_16:29:00_first.jpg
    Processing file: frame_16:29:00_last.jpg
    Processing file: frame_16:30:00_first.jpg
    Processing file: frame_16:30:00_last.jpg
    Processing file: frame_16:31:00_first.jpg
    Processing file: frame_16:31:00_last.jpg
    Processing file: frame_16:32:00_first.jpg
    Processing file: frame_16:32:00_last.jpg
    Processing file: frame_16:33:00_first.jpg
    Processing file: frame_16:33:00_last.jpg
    Processing file: frame_16:34:00_first.jpg
    Processing file: frame_16:34:00_last.jpg
    Processing file: frame_16:35:00_first.jpg
    Processing file: frame_16:35:00_last.jpg
    Processing file: frame_16:36:00_first.jpg
    Processing file: frame_16:36:00_last.jpg
    Processing file: frame_16:37:00_first.jpg
    Processing file: frame_16:37:00_last.jpg
    Processing file: frame_16:38:00_first.jpg
    Processing file: frame_16:38:00_last.jpg
    Processing file: frame_16:39:00_first.jpg
    Processing file: frame_16:39:00_last.jpg
    Processing file: frame_16:40:00_first.jpg
    Processing file: frame_16:40:00_last.jpg
    Processing file: frame_16:41:00_first.jpg
    Processing file: frame_16:41:00_last.jpg
    Processing file: frame_16:42:00_first.jpg
    Processing file: frame_16:42:00_last.jpg
```

```
Processing file: frame_16:42:00_last.jpg
Processing file: frame_16:43:00_first.jpg
Processing file: frame_16:43:00_last.jpg
Processing file: frame_16:44:00_first.jpg
Processing file: frame_16:44:00_last.jpg
Processing file: frame_16:45:00_first.jpg
Processing file: frame_16:45:00_last.jpg
Processing file: frame_16:46:00_first.jpg
Processing file: frame_16:46:00_last.jpg
Processing file: frame_16:47:00_first.jpg
Processing file: frame_16:47:00_last.jpg
Processing file: frame_16:48:00_first.jpg
Processing file: frame_16:48:00_last.jpg
Processing file: frame_16:49:00_first.jpg
Processing file: frame_16:49:00_last.jpg
Processing file: frame_16:50:00_first.jpg
Processing file: frame_16:50:00_last.jpg
Processing file: frame_16:51:00_first.jpg
Processing file: frame_16:51:00_last.jpg
Processing file: frame_16:52:00_first.jpg
Processing file: frame_16:52:00_last.jpg
Processing file: frame_16:53:00_first.jpg
Processing file: frame_16:53:00_last.jpg
Processing file: frame_16:54:00_first.jpg
Processing file: frame_16:54:00_last.jpg
Processing file: frame_16:55:00_first.jpg
Processing file: frame_16:55:00_last.jpg
Processing file: frame_16:56:00_first.jpg
Processing file: frame_16:56:00_last.jpg
Processing file: frame_16:57:00_first.jpg
Processing file: frame_16:57:00_last.jpg
Processing file: frame_16:58:00_first.jpg
Processing file: frame_16:58:00_last.jpg
Processing file: frame_16:59:00_first.jpg
Processing file: frame_16:59:00_last.jpg
Processing file: frame_17:00:00_first.jpg
Processing file: frame_17:00:00_last.jpg
Processing file: frame_17:01:00_first.jpg
Processing file: frame_17:01:00_last.jpg
Processing file: frame_17:02:00_first.jpg
Processing file: frame_17:02:00_last.jpg
Processing file: frame_17:03:00_first.jpg
Processing file: frame_17:03:00_last.jpg
Processing file: frame_17:04:00_first.jpg
Processing file: frame_17:04:00_last.jpg
Processing file: frame_17:05:00_first.jpg
Processing file: frame_17:05:00_last.jpg
Processing file: frame_17:06:00_first.jpg
Processing file: frame_17:06:00_last.jpg
Processing file: frame_17:07:00_first.jpg
Processing file: frame_17:07:00_last.jpg
Processing file: frame_17:08:00_first.jpg
Processing file: frame_17:08:00_last.jpg
Processing file: frame_17:09:00_first.jpg
Processing file: frame_17:09:00_last.jpg
Processing file: frame_17:10:00_first.jpg
Processing file: frame_17:10:00_last.jpg
Processing file: frame_17:11:00_first.jpg
Processing file: frame_17:11:00_last.jpg
Processing file: frame_17:12:00_first.jpg
Processing file: frame_17:12:00_last.jpg
Processing file: frame_17:13:00_first.jpg
Processing file: frame_17:13:00_last.jpg
Processing file: frame_17:14:00_first.jpg
Processing file: frame_17:14:00_last.jpg
Processing file: frame_17:15:00_first.jpg
Processing file: frame_17:15:00_last.jpg
Processing file: frame_17:16:00_first.jpg
Processing file: frame_17:16:00_last.jpg
Processing file: frame_17:17:00_first.jpg
Processing file: frame_17:17:00_last.jpg
Processing file: frame_17:18:00_first.jpg
Processing file: frame_17:18:00_last.jpg
Processing file: frame_17:19:00_first.jpg
Processing file: frame_17:19:00_last.jpg
Processing file: frame_17:20:00_first.jpg
Processing file: frame_17:20:00_last.jpg
Processing file: frame_17:21:00_first.jpg
Processing file: frame_17:21:00_last.jpg
Processing file: frame_17:22:00_first.jpg
Processing file: frame_17:22:00_last.jpg
Processing file: frame_17:23:00_first.jpg
Processing file: frame_17:23:00_last.jpg
Processing file: frame_17:24:00_first.jpg
Processing file: frame_17:24:00_last.jpg
Processing file: frame_17:25:00_first.jpg
Processing file: frame_17:25:00_last.jpg
Processing file: frame_17:26:00_first.jpg
Processing file: frame_17:26:00_last.jpg
Processing file: frame_17:27:00_first.jpg
Processing file: frame_17:27:00_last.jpg
Processing file: frame_17:28:00_first.jpg
Processing file: frame_17:28:00_last.jpg
Processing file: frame_17:29:00_first.jpg
Processing file: frame_17:29:00_last.jpg
Processing file: frame_17:30:00_first.jpg
Processing file: frame_17:30:00_last.jpg
Processing file: frame_17:31:00_first.jpg
Processing file: frame_17:31:00_last.jpg
Processing file: frame_17:32:00_first.jpg
Processing file: frame_17:32:00_last.jpg
Processing file: frame_17:33:00_first.jpg
Processing file: frame_17:33:00_first.jpg
Processing file: frame_17:33:00_last.jpg
Processing file: frame_17:34:00_first.jpg
Processing file: frame_17:34:00_last.jpg
Processing file: frame_17:35:00_first.jpg
Processing file: frame_17:35:00_last.jpg
Processing file: frame_17:36:00_first.jpg
Processing file: frame_17:36:00_last.jpg
Processing file: frame_17:37:00_first.jpg
Processing file: frame_17:37:00_last.jpg
Processing file: frame_17:38:00_first.jpg
Processing file: frame_17:38:00_last.jpg
Processing file: frame_17:39:00_first.jpg
Processing file: frame_17:39:00_last.jpg
Processing file: frame_17:40:00_first.jpg
Processing file: frame_17:40:00_last.jpg
Processing file: frame_17:41:00_first.jpg
Processing file: frame_17:41:00_last.jpg
Processing file: frame_17:42:00_first.jpg
Processing file: frame_17:42:00_last.jpg
Processing file: frame_17:43:00_first.jpg
Processing file: frame_17:43:00_last.jpg
Processing file: frame_17:44:00_first.jpg
Processing file: frame_17:44:00_last.jpg
Processing file: frame_17:45:00_first.jpg
Processing file: frame_17:45:00_last.jpg
Processing file: frame_17:46:00_first.jpg
Processing file: frame_17:46:00_last.jpg
Processing file: frame_17:47:00_first.jpg
Processing file: frame_17:47:00_last.jpg
Processing file: frame_17:48:00_first.jpg
Processing file: frame_17:48:00_last.jpg
Processing file: frame_17:49:00_first.jpg
Processing file: frame_17:49:00_last.jpg
Processing file: frame_17:50:00_first.jpg
Processing file: frame_17:50:00_last.jpg
Processing file: frame_17:51:00_first.jpg
Processing file: frame_17:51:00_last.jpg
Processing file: frame_17:52:00_first.jpg
Processing file: frame_17:52:00_last.jpg
Processing file: frame_17:53:00_first.jpg
Processing file: frame_17:53:00_last.jpg
Processing file: frame_17:54:00_first.jpg
Processing file: frame_17:54:00_last.jpg
Processing file: frame_17:55:00_first.jpg
Processing file: frame_17:55:00_last.jpg
Processing file: frame_17:56:00_first.jpg
Processing file: frame_17:56:00_last.jpg
Processing file: frame_17:57:00_first.jpg
Processing file: frame_17:57:00_last.jpg
Processing file: frame_17:58:00_first.jpg
Processing file: frame_17:58:00_last.jpg
Processing file: frame_17:59:00_first.jpg
Processing file: frame_17:59:00_last.jpg
Processing file: frame_18:00:00_first.jpg
Processing file: frame_18:00:00_last.jpg
Processing file: frame_18:01:00_first.jpg
Processing file: frame_18:01:00_last.jpg
Processing file: frame_18:02:00_first.jpg
Processing file: frame_18:02:00_last.jpg
Processing file: frame_18:03:00_first.jpg
Processing file: frame_18:03:00_last.jpg
Processing file: frame_18:04:00_first.jpg
Processing file: frame_18:04:00_last.jpg
Processing file: frame_18:05:00_first.jpg
Processing file: frame_18:05:00_last.jpg
Processing file: frame_18:06:00_first.jpg
Processing file: frame_18:06:00_last.jpg
Processing file: frame_18:07:00_first.jpg
Processing file: frame_18:07:00_last.jpg
Processing file: frame_18:08:00_first.jpg
Processing file: frame_18:08:00_last.jpg
Processing file: frame_18:09:00_first.jpg
Processing file: frame_18:09:00_last.jpg
Processing file: frame_18:10:00_first.jpg
Processing file: frame_18:10:00_last.jpg
Processing file: frame_18:11:00_first.jpg
Processing file: frame_18:11:00_last.jpg
Processing file: frame_18:12:00_first.jpg
Processing file: frame_18:12:00_last.jpg
Processing file: frame_18:13:00_first.jpg
Processing file: frame_18:13:00_last.jpg
Processing file: frame_18:14:00_first.jpg
Processing file: frame_18:14:00_last.jpg
Processing file: frame_18:15:00_first.jpg
Processing file: frame_18:15:00_last.jpg
Processing file: frame_18:16:00_first.jpg
Processing file: frame_18:16:00_last.jpg
Processing file: frame_18:17:00_first.jpg
Processing file: frame_18:17:00_last.jpg
Processing file: frame_18:18:00_first.jpg
Processing file: frame_18:18:00_last.jpg
Processing file: frame_18:19:00_first.jpg
Processing file: frame_18:19:00_last.jpg
Processing file: frame_18:20:00_first.jpg
Processing file: frame_18:20:00_last.jpg
Processing file: frame_18:21:00_first.jpg
Processing file: frame_18:21:00_last.jpg
Processing file: frame_18:22:00_first.jpg
Processing file: frame_18:22:00_last.jpg
Processing file: frame_18:23:00_first.jpg
Processing file: frame_18:23:00_last.jpg
Processing file: frame_18:23:00_last.jpg
Processing file: frame_18:24:00_first.jpg
Processing file: frame_18:24:00_last.jpg
Processing file: frame_18:25:00_first.jpg
Processing file: frame_18:25:00_last.jpg
Processing file: frame_18:26:00_first.jpg
Processing file: frame_18:26:00_last.jpg
Processing file: frame_18:27:00_first.jpg
Processing file: frame_18:27:00_last.jpg
Processing file: frame_18:28:00_first.jpg
Processing file: frame_18:28:00_last.jpg
Processing file: frame_18:29:00_first.jpg
Processing file: frame_18:29:00_last.jpg
Processing file: frame_18:30:00_first.jpg
Processing file: frame_18:30:00_last.jpg
Processing file: frame_18:31:00_first.jpg
Processing file: frame_18:31:00_last.jpg
Processing file: frame_18:32:00_first.jpg
Processing file: frame_18:32:00_last.jpg
Processing file: frame_18:33:00_first.jpg
Processing file: frame_18:33:00_last.jpg
Processing file: frame_18:34:00_first.jpg
Processing file: frame_18:34:00_last.jpg
Processing file: frame_18:35:00_first.jpg
Processing file: frame_18:35:00_last.jpg
Processing file: frame_18:36:00_first.jpg
Processing file: frame_18:36:00_last.jpg
Processing file: frame_18:37:00_first.jpg
Processing file: frame_18:37:00_last.jpg
Processing file: frame_18:38:00_first.jpg
Processing file: frame_18:38:00_last.jpg
Processing file: frame_18:39:00_first.jpg
Processing file: frame_18:39:00_last.jpg
Processing file: frame_18:40:00_first.jpg
Processing file: frame_18:40:00_last.jpg
Processing file: frame_18:41:00_first.jpg
Processing file: frame_18:41:00_last.jpg
Processing file: frame_18:42:00_first.jpg
Processing file: frame_18:42:00_last.jpg
Processing file: frame_18:43:00_first.jpg
Processing file: frame_18:43:00_last.jpg
Processing file: frame_18:44:00_first.jpg
Processing file: frame_18:44:00_last.jpg
Processing file: frame_18:45:00_first.jpg
Processing file: frame_18:45:00_last.jpg
Processing file: frame_18:46:00_first.jpg
Processing file: frame_18:46:00_last.jpg
Processing file: frame_18:47:00_first.jpg
Processing file: frame_18:47:00_last.jpg
Processing file: frame_18:48:00_first.jpg
Processing file: frame_18:48:00_last.jpg
```

```
    Processing file: frame_18:49:00_first.jpg
    Processing file: frame_18:49:00_last.jpg
    Processing file: frame_18:50:00_first.jpg
    Processing file: frame_18:50:00_last.jpg
    Processing file: frame_18:51:00_first.jpg
    Processing file: frame_18:51:00_last.jpg
    Processing file: frame_18:52:00_first.jpg
    Processing file: frame_18:52:00_last.jpg
    Processing file: frame_18:53:00_first.jpg
    Processing file: frame_18:53:00_last.jpg
    Processing file: frame_18:54:00_first.jpg
    Processing file: frame_18:54:00_last.jpg
    Processing file: frame_18:55:00_first.jpg
    Processing file: frame_18:55:00_last.jpg
    Processing file: frame_18:56:00_first.jpg
    Processing file: frame_18:56:00_last.jpg
    Processing file: frame_18:57:00_first.jpg
    Processing file: frame_18:57:00_last.jpg
    Processing file: frame_18:58:00_first.jpg
    Processing file: frame_18:58:00_last.jpg
    Processing file: frame_18:59:00_first.jpg
    Processing file: frame_18:59:00_last.jpg
    Processing file: frame_19:00:00_first.jpg
    Processing file: frame_19:00:00_last.jpg
    Processing file: frame_19:01:00_first.jpg
    Processing file: frame_19:01:00_last.jpg
    Processing file: frame_19:02:00_first.jpg
    Processing file: frame_19:02:00_last.jpg
    Processing file: frame_19:03:00_first.jpg
    Processing file: frame_19:03:00_last.jpg
    Processing file: frame_19:04:00_first.jpg
    Processing file: frame_19:04:00_last.jpg
    Processing file: frame_19:05:00_first.jpg
    Processing file: frame_19:05:00_last.jpg
    Processing file: frame_19:06:00_first.jpg
    Processing file: frame_19:06:00_last.jpg
    Processing file: frame_19:07:00_first.jpg
    Processing file: frame_19:07:00_last.jpg
    Processing file: frame_19:08:00_first.jpg
    Processing file: frame_19:08:00_last.jpg
    Processing file: frame_19:09:00_first.jpg
    Processing file: frame_19:09:00_last.jpg
    Processing file: frame_19:10:00_first.jpg
    Processing file: frame_19:10:00_last.jpg
    Processing file: frame_19:11:00_first.jpg
    Processing file: frame_19:11:00_last.jpg
    Processing file: frame_19:12:00_first.jpg
    Processing file: frame_19:12:00_last.jpg
    Processing file: frame_19:13:00_first.jpg
    Processing file: frame_19:13:00_last.jpg
```

```
Processing file: frame_19:14:00_first.jpg
Processing file: frame_19:14:00_last.jpg
Processing file: frame_19:15:00_first.jpg
Processing file: frame_19:15:00_last.jpg
Processing file: frame_19:16:00_first.jpg
Processing file: frame_19:16:00_last.jpg
Processing file: frame_19:17:00_first.jpg
Processing file: frame_19:17:00_last.jpg
Processing file: frame_19:18:00_first.jpg
Processing file: frame_19:18:00_last.jpg
Processing file: frame_19:19:00_first.jpg
Processing file: frame_19:19:00_last.jpg
Processing file: frame_19:20:00_first.jpg
Processing file: frame_19:20:00_last.jpg
Processing file: frame_19:21:00_first.jpg
Processing file: frame_19:21:00_last.jpg
Processing file: frame_19:22:00_first.jpg
Processing file: frame_19:22:00_last.jpg
Processing file: frame_19:23:00_first.jpg
Processing file: frame_19:23:00_last.jpg
Processing file: frame_19:24:00_first.jpg
Processing file: frame_19:24:00_last.jpg
Processing file: frame_19:25:00_first.jpg
Processing file: frame_19:25:00_last.jpg
Processing file: frame_19:26:00_first.jpg
Processing file: frame_19:26:00_last.jpg
Processing file: frame_19:27:00_first.jpg
Processing file: frame_19:27:00_last.jpg
Processing file: frame_19:28:00_first.jpg
Processing file: frame_19:28:00_last.jpg
Processing file: frame_19:29:00_first.jpg
Processing file: frame_19:29:00_last.jpg
Processing file: frame_19:30:00_first.jpg
Processing file: frame_19:30:00_last.jpg
Processing file: frame_19:31:00_first.jpg
Processing file: frame_19:31:00_last.jpg
Processing file: frame_19:32:00_first.jpg
Processing file: frame_19:32:00_last.jpg
Processing file: frame_19:33:00_first.jpg
Processing file: frame_19:33:00_last.jpg
Processing file: frame_19:34:00_first.jpg
Processing file: frame_19:34:00_last.jpg
Processing file: frame_19:35:00_first.jpg
Processing file: frame_19:35:00_last.jpg
Processing file: frame_19:36:00_first.jpg
Processing file: frame_19:36:00_last.jpg
Processing file: frame_19:37:00_first.jpg
Processing file: frame_19:37:00_last.jpg
Processing file: frame_19:38:00_first.jpg
Processing file: frame_19:38:00_last.jpg
Processing file: frame_19:39:00_first.jpg
Processing file: frame_19:39:00_first.jpg
Processing file: frame_19:39:00_last.jpg
Processing file: frame_19:40:00_first.jpg
Processing file: frame_19:40:00_last.jpg
Processing file: frame_19:41:00_first.jpg
Processing file: frame_19:41:00_last.jpg
Processing file: frame_19:42:00_first.jpg
Processing file: frame_19:42:00_last.jpg
Processing file: frame_19:43:00_first.jpg
Processing file: frame_19:43:00_last.jpg
Processing file: frame_19:44:00_first.jpg
Processing file: frame_19:44:00_last.jpg
Processing file: frame_19:45:00_first.jpg
Processing file: frame_19:45:00_last.jpg
Processing file: frame_19:46:00_first.jpg
Processing file: frame_19:46:00_last.jpg
Processing file: frame_19:47:00_first.jpg
Processing file: frame_19:47:00_last.jpg
Processing file: frame_19:48:00_first.jpg
Processing file: frame_19:48:00_last.jpg
Processing file: frame_19:49:00_first.jpg
Processing file: frame_19:49:00_last.jpg
Processing file: frame_19:50:00_first.jpg
Processing file: frame_19:50:00_last.jpg
Processing file: frame_19:51:00_first.jpg
Processing file: frame_19:51:00_last.jpg
Processing file: frame_19:52:00_first.jpg
Processing file: frame_19:52:00_last.jpg
Processing file: frame_19:53:00_first.jpg
Processing file: frame_19:53:00_last.jpg
Processing file: frame_19:54:00_first.jpg
Processing file: frame_19:54:00_last.jpg
Processing file: frame_19:55:00_first.jpg
Processing file: frame_19:55:00_last.jpg
Processing file: frame_19:56:00_first.jpg
Processing file: frame_19:56:00_last.jpg
Processing file: frame_19:57:00_first.jpg
Processing file: frame_19:57:00_last.jpg
Processing file: frame_19:58:00_first.jpg
Processing file: frame_19:58:00_last.jpg
Processing file: frame_19:59:00_first.jpg
Processing file: frame_19:59:00_last.jpg
Processing file: frame_20:00:00_first.jpg
Processing file: frame_20:00:00_last.jpg
Processing file: frame_20:01:00_first.jpg
Processing file: frame_20:01:00_last.jpg
Processing file: frame_20:02:00_first.jpg
Processing file: frame_20:02:00_last.jpg
Processing file: frame_20:03:00_first.jpg
Processing file: frame_20:03:00_last.jpg
Processing file: frame_20:04:00_first.jpg
Processing file: frame_20:04:00_last.jpg
Processing file: frame_20:05:00_first.jpg
Processing file: frame_20:05:00_last.jpg
Processing file: frame_20:06:00_first.jpg
Processing file: frame_20:06:00_last.jpg
Processing file: frame_20:07:00_first.jpg
Processing file: frame_20:07:00_last.jpg
Processing file: frame_20:08:00_first.jpg
Processing file: frame_20:08:00_last.jpg
Processing file: frame_20:09:00_first.jpg
Processing file: frame_20:09:00_last.jpg
Processing file: frame_20:10:00_first.jpg
Processing file: frame_20:10:00_last.jpg
Processing file: frame_20:11:00_first.jpg
Processing file: frame_20:11:00_last.jpg
Processing file: frame_20:12:00_first.jpg
Processing file: frame_20:12:00_last.jpg
Processing file: frame_20:13:00_first.jpg
Processing file: frame_20:13:00_last.jpg
Processing file: frame_20:14:00_first.jpg
Processing file: frame_20:14:00_last.jpg
Processing file: frame_20:15:00_first.jpg
Processing file: frame_20:15:00_last.jpg
Processing file: frame_20:16:00_first.jpg
Processing file: frame_20:16:00_last.jpg
Processing file: frame_20:17:00_first.jpg
Processing file: frame_20:17:00_last.jpg
Processing file: frame_20:18:00_first.jpg
Processing file: frame_20:18:00_last.jpg
Processing file: frame_20:19:00_first.jpg
Processing file: frame_20:19:00_last.jpg
Processing file: frame_20:20:00_first.jpg
Processing file: frame_20:20:00_last.jpg
Processing file: frame_20:21:00_first.jpg
Processing file: frame_20:21:00_last.jpg
Processing file: frame_20:22:00_first.jpg
Processing file: frame_20:22:00_last.jpg
Processing file: frame_20:23:00_first.jpg
Processing file: frame_20:23:00_last.jpg
Processing file: frame_20:24:00_first.jpg
Processing file: frame_20:24:00_last.jpg
Processing file: frame_20:25:00_first.jpg
Processing file: frame_20:25:00_last.jpg
Processing file: frame_20:26:00_first.jpg
Processing file: frame_20:26:00_last.jpg
Processing file: frame_20:27:00_first.jpg
Processing file: frame_20:27:00_last.jpg
Processing file: frame_20:28:00_first.jpg
Processing file: frame_20:28:00_last.jpg
Processing file: frame_20:29:00_first.jpg
Processing file: frame_20:29:00_last.jpg
Processing file: frame_20:29:00_last.jpg
Processing file: frame_20:30:00_first.jpg
Processing file: frame_20:30:00_last.jpg
Processing file: frame_20:31:00_first.jpg
Processing file: frame_20:31:00_last.jpg
Processing file: frame_20:32:00_first.jpg
Processing file: frame_20:32:00_last.jpg
Processing file: frame_20:33:00_first.jpg
Processing file: frame_20:33:00_last.jpg
Processing file: frame_20:34:00_first.jpg
Processing file: frame_20:34:00_last.jpg
Processing file: frame_20:35:00_first.jpg
Processing file: frame_20:35:00_last.jpg
Processing file: frame_20:36:00_first.jpg
Processing file: frame_20:36:00_last.jpg
Processing file: frame_20:37:00_first.jpg
Processing file: frame_20:37:00_last.jpg
Processing file: frame_20:38:00_first.jpg
Processing file: frame_20:38:00_last.jpg
Processing file: frame_20:39:00_first.jpg
Processing file: frame_20:39:00_last.jpg
Processing file: frame_20:40:00_first.jpg
Processing file: frame_20:40:00_last.jpg
Processing file: frame_20:41:00_first.jpg
Processing file: frame_20:41:00_last.jpg
Processing file: frame_20:42:00_first.jpg
Processing file: frame_20:42:00_last.jpg
Processing file: frame_20:43:00_first.jpg
Processing file: frame_20:43:00_last.jpg
Processing file: frame_20:44:00_first.jpg
Processing file: frame_20:44:00_last.jpg
Processing file: frame_20:45:00_first.jpg
Processing file: frame_20:45:00_last.jpg
Processing file: frame_20:46:00_first.jpg
Processing file: frame_20:46:00_last.jpg
Processing file: frame_20:47:00_first.jpg
Processing file: frame_20:47:00_last.jpg
Processing file: frame_20:48:00_first.jpg
Processing file: frame_20:48:00_last.jpg
Processing file: frame_20:49:00_first.jpg
Processing file: frame_20:49:00_last.jpg
Processing file: frame_20:50:00_first.jpg
Processing file: frame_20:50:00_last.jpg
Processing file: frame_20:51:00_first.jpg
Processing file: frame_20:51:00_last.jpg
Processing file: frame_20:52:00_first.jpg
Processing file: frame_20:52:00_last.jpg
Processing file: frame_20:53:00_first.jpg
Processing file: frame_20:53:00_last.jpg
Processing file: frame_20:54:00_first.jpg
Processing file: frame_20:54:00_last.jpg
Processing file: frame_20:55:00_first.jpg
Processing file: frame_20:55:00_last.jpg
Processing file: frame_20:56:00_first.jpg
Processing file: frame_20:56:00_last.jpg
Processing file: frame_20:57:00_first.jpg
Processing file: frame_20:57:00_last.jpg
Processing file: frame_20:58:00_first.jpg
Processing file: frame_20:58:00_last.jpg
Processing file: frame_20:59:00_first.jpg
Processing file: frame_20:59:00_last.jpg
Processing file: frame_21:00:00_first.jpg
Processing file: frame_21:00:00_last.jpg
Processing file: frame_21:01:00_first.jpg
Processing file: frame_21:01:00_last.jpg
Processing file: frame_21:02:00_first.jpg
Processing file: frame_21:02:00_last.jpg
Processing file: frame_21:03:00_first.jpg
Processing file: frame_21:03:00_last.jpg
Processing file: frame_21:04:00_first.jpg
Processing file: frame_21:04:00_last.jpg
Processing file: frame_21:05:00_first.jpg
Processing file: frame_21:05:00_last.jpg
Processing file: frame_21:06:00_first.jpg
Processing file: frame_21:06:00_last.jpg
Processing file: frame_21:07:00_first.jpg
Processing file: frame_21:07:00_last.jpg
Processing file: frame_21:08:00_first.jpg
Processing file: frame_21:08:00_last.jpg
Processing file: frame_21:09:00_first.jpg
Processing file: frame_21:09:00_last.jpg
Processing file: frame_21:10:00_first.jpg
Processing file: frame_21:10:00_last.jpg
Processing file: frame_21:11:00_first.jpg
Processing file: frame_21:11:00_last.jpg
Processing file: frame_21:12:00_first.jpg
Processing file: frame_21:12:00_last.jpg
Processing file: frame_21:13:00_first.jpg
Processing file: frame_21:13:00_last.jpg
Processing file: frame_21:14:00_first.jpg
Processing file: frame_21:14:00_last.jpg
Processing file: frame_21:15:00_first.jpg
Processing file: frame_21:15:00_last.jpg
Processing file: frame_21:16:00_first.jpg
Processing file: frame_21:16:00_last.jpg
Processing file: frame_21:17:00_first.jpg
Processing file: frame_21:17:00_last.jpg
Processing file: frame_21:18:00_first.jpg
Processing file: frame_21:18:00_last.jpg
Processing file: frame_21:19:00_first.jpg
Processing file: frame_21:19:00_last.jpg
Processing file: frame_21:20:00_first.jpg
Processing file: frame_21:20:00_last.jpg
Processing file: frame_21:21:00_first.jpg
Processing file: frame_21:21:00_last.jpg
Results saved to /content/drive/MyDrive/Fire Detection/model_evaluation_result
Models saved successfully.
```



GUI

```
pip install gradio torch torchvision pandas openpyxl
```

```
Requirement already satisfied: gradio in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: torchvision in /usr/local/lib/python3.10/dist-p
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: openpyxl in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: aiofiles<24.0,>=22.0 in /usr/local/lib/python3.
Requirement already satisfied: anyio<5.0,>=3.0 in /usr/local/lib/python3.10/di
Requirement already satisfied: fastapi<1.0,>=0.115.2 in /usr/local/lib/python3
Requirement already satisfied: ffmpy in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: gradio-client==1.4.3 in /usr/local/lib/python3.
Requirement already satisfied: httpx>=0.24.1 in /usr/local/lib/python3.10/dist
Requirement already satisfied: huggingface-hub>=0.25.1 in /usr/local/lib/pytho
```

```
    Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.10/dist-pa
    Requirement already satisfied: markupsafe~=2.0 in /usr/local/lib/python3.10/di
    Requirement already satisfied: numpy<3.0,>=1.0 in /usr/local/lib/python3.10/di
    Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.10/dist-p
    Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-pac
    Requirement already satisfied: pillow<12.0,>=8.0 in /usr/local/lib/python3.10/
    Requirement already satisfied: pydantic>=2.0 in /usr/local/lib/python3.10/dist
    Requirement already satisfied: pydub in /usr/local/lib/python3.10/dist-package
    Requirement already satisfied: python-multipart==0.0.12 in /usr/local/lib/pyth
    Requirement already satisfied: pyyaml<7.0,>=5.0 in /usr/local/lib/python3.10/d
    Requirement already satisfied: ruff>=0.2.2 in /usr/local/lib/python3.10/dist-p
    Requirement already satisfied: safehttpx<1.0,>=0.1.1 in /usr/local/lib/python3
    Requirement already satisfied: semantic-version~=2.0 in /usr/local/lib/python3
    Requirement already satisfied: starlette<1.0,>=0.40.0 in /usr/local/lib/python
    Requirement already satisfied: tomlkit==0.12.0 in /usr/local/lib/python3.10/di
    Requirement already satisfied: typer<1.0,>=0.12 in /usr/local/lib/python3.10/d
    Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/python
    Requirement already satisfied: uvicorn>=0.14.0 in /usr/local/lib/python3.10/di
    Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packag
    Requirement already satisfied: websockets<13.0,>=10.0 in /usr/local/lib/python
    Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-pack
    Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-pack
    Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist
    Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10
    Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python
    Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dis
    Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.10/dist-pa
    Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.10/dist-pac
    Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dis
    Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packa
    Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.10/dist
    Requirement already satisfied: h11<0.15,>=0.13 in /usr/local/lib/python3.10/di
    Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-pack
    Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python
    Requirement already satisfied: pydantic-core==2.23.4 in /usr/local/lib/python3
    Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-pack
    Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.10
    Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.10/dist
    Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3
    Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/pytho
    Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/pyth
    Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10
    Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-pa
```

```
import gradio as gr
import pandas as pd
import pandas as pd
import numpy as np
import cv2
import joblib
import matplotlib.pyplot as plt
import tempfile
import base64

# Load the pre-trained models
model_ce = joblib.load('/content/drive/MyDrive/Fire Detection/model_ce.joblib')
model_dre = joblib.load('/content/drive/MyDrive/Fire Detection/model_dre.joblib')

# Load images and convert them to base64
def image_to_base64(image_path):
    with open(image_path, "rb") as img_file:
        return base64.b64encode(img_file.read()).decode("utf-8")

university_logo_base64 = image_to_base64('/content/drive/MyDrive/Fire Detection/uni
adnoc_logo_base64 = image_to_base64('/content/drive/MyDrive/Fire Detection/adnoc_lo

def extract_kpis(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    contours, _ = cv2.findContours(gray, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE
    angle = cv2.minAreaRect(contours[0])[-1] if contours else 0
    ratio = (cv2.boundingRect(contours[0])[2] / cv2.boundingRect(contours[0])[3]) i
    orientation = (cv2.phase(np.array([cv2.moments(contours[0])["m10"]]), np.array(
    color = image.mean(axis=0).mean(axis=0)[0]
    hist = cv2.calcHist([gray], [0], None, [256], [0, 256]).flatten()
    return {'angle': angle, 'ratio': ratio, 'orientation': orientation, 'color': co

def upload_excel(file):
    try:
        df = pd.read_excel(file)
        for col in df.select_dtypes(include=['datetime64']).columns:
            df[col] = df[col].fillna(pd.Timestamp("1970-01-01")).astype(int) // 10*
        df = df.apply(lambda col: col.fillna(0) if col.dtype in ['float64', 'int64'
        last_row = df.iloc[-1].to_dict()
        return last_row
    except Exception as e:
        return {"Error": str(e)}

def combine_data(kpi_data, excel_row):
    try:
        combined_data = {**kpi_data, **excel_row}
        expected_feature_names = model_ce.feature_names_in_
        for feature in expected_feature_names:
```

```
        for feature in expected_feature_names:
            if feature not in combined_data:
                combined_data[feature] = 0
        ordered_combined_data = [combined_data[feature] for feature in expected_fea
        return ordered_combined_data
    except Exception as e:
        return {"Error": str(e)}

def predict_ce_dre(combined_data):
    try:
        ce_prediction = model_ce.predict([combined_data])[0]
        dre_prediction = model_dre.predict([combined_data])[0]
        return f"Predicted CE: {ce_prediction:.2f}, Predicted DRE: {dre_prediction:
    except Exception as e:
        return f"Prediction Error: {str(e)}"

def full_pipeline(image, excel_file):
    try:
        kpi_data = extract_kpis(image)
        excel_row = upload_excel(excel_file)
        if "Error" in excel_row:
            return excel_row["Error"], None
        combined_data = combine_data(kpi_data, excel_row)
        actual_ce = excel_row.get("CE", 0)
        actual_dre = excel_row.get("DRE", 0)
        ce_prediction = model_ce.predict([combined_data])[0]
        dre_prediction = model_dre.predict([combined_data])[0]
        with tempfile.NamedTemporaryFile(suffix=".png", delete=False) as temp_file:
            fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 4))
            ax1.bar(['Actual CE', 'Predicted CE'], [actual_ce, ce_prediction], colo
            ax1.set_title('CE: Actual vs Predicted')
            ax1.set_ylabel('Value')
            ax2.bar(['Actual DRE', 'Predicted DRE'], [actual_dre, dre_prediction],
            ax2.set_title('DRE: Actual vs Predicted')
            ax2.set_ylabel('Value')
            plt.tight_layout()
            plt.savefig(temp_file.name)
            plt.close(fig)
            result_text = (
                f"Actual CE: {actual_ce}, Predicted CE: {ce_prediction:.2f}\n"
                f"Actual DRE: {actual_dre}, Predicted DRE: {dre_prediction:.2f}"
            )
            return result_text, temp_file.name
    except Exception as e:
        return f"Error in pipeline: {str(e)}", None
```

```
# Define the Gradio interface
with gr.Blocks() as interface:
    gr.HTML(
        f"""
        <div style="text-align: center; padding: 20px; background-color: #f0f0f5;">
            <img src="data:image/jpg;base64,{university_logo_base64}" alt="Universi
            <h2 style="display:inline-block; color: #004d99;">Quantifying the Flare
            <img src="data:image/jpg;base64,{adnoc_logo_base64}" alt="ADNOC Logo" w
        </div>
        """
    )

    gr.Markdown("<p style='text-align: center;'>Upload a flare image and the Excel

    with gr.Accordion("Step 1: Upload Image and Extract Flaring KPIs", open=True):
        with gr.Row():
            image_input = gr.Image(label="Upload Flare Image", type="numpy")
            kpi_output = gr.JSON(label="Extracted Flaring KPIs")
            extract_button = gr.Button("Extract Flaring KPIs", variant="primary")
        extract_button.click(extract_kpis, inputs=image_input, outputs=kpi_output)

    with gr.Accordion("Step 2: Upload Excel and Select Row", open=False):
        with gr.Row():
            excel_input = gr.File(label="Upload Excel File")
            excel_output = gr.JSON(label="Excel Data Row")
            upload_button = gr.Button("Select the Row", variant="primary")
        upload_button.click(upload_excel, inputs=excel_input, outputs=excel_output)

    with gr.Accordion("Step 3: Combine Data and Predict", open=False):
        with gr.Row():
            combine_button = gr.Button("Combine Data", variant="secondary")
            combined_data_output = gr.JSON(label="Combined Feature Data")
            predict_button = gr.Button("Predict CE & DRE", variant="primary")
            prediction_output = gr.Textbox(label="Prediction Results")
        combine_button.click(combine_data, inputs=[kpi_output, excel_output], outpu
        predict_button.click(predict_ce_dre, inputs=combined_data_output, outputs=p

    with gr.Accordion("Step 4: Full Pipeline Execution", open=False):
        with gr.Row():
            full_pipeline_button = gr.Button("Run Full Pipeline", variant="primary"
            full_pipeline_output = gr.Textbox(label="Results")
            full_pipeline_graph = gr.Image(label="CE & DRE Comparison Plot")
        full_pipeline_button.click(full_pipeline, inputs=[image_input, excel_input]

interface.launch()
```

```
Running Gradio in a Colab notebook requires sharing enabled. Automatically set

Colab notebook detected. To show errors in colab notebook, set debug=True in l
* Running on public URL: https://80bdb398a31d3de826.gradio.live

This share link expires in 72 hours. For free permanent hosting and GPU upgrad
```

The page you requested was not found.

Sorry, the page you are looking for is currently unavailable.
Please try again later.

The server is powered by frp.

Faithfully yours, frp.