```
import pandas as pd

# Load the dataset
file_path = '/mnt/data/image_labels (1).csv'
df = pd.read_csv(file_path)

# Display the first few rows and summary info
df.head(), df.info()
```

Task

Tell me about this dataset.

Here is all the data you need: "image_labels (1).csv"

Data loading

Subtask:

Load the data from "image_labels (1).csv" into a pandas DataFrame.

Reasoning: Load the data from "image_labels (1).csv" into a pandas DataFrame and display the first few rows.

```
import pandas as pd

try:
    df = pd.read_csv('image_labels (1).csv')
    display(df.head())
except FileNotFoundError:
    print("Error: 'image_labels (1).csv' not found. Please ensure the file exists in the current directory.")
    df = None  # Set df to None to indicate failure
except Exception as e:
    print(f"An unexpected error occurred: {e}")
    df = None
```

```
{"summary":"{\n  \"name\": \"    df = None\",\n  \"rows\": 5,\n  \"fields\": [\n    {\n      \"column\": \"filename\",\n      \"properties\": {\n        \"dtype\": \"string\",\n        \"num_unique_values\": 5,\n        \"samples\": [\n          \"0/0.053.1.augmented.png\",\n          \"0/0.042.3.augmented.png\",\n          \"0/0.052.4.augmented.png\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"label\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 1,\n        \"samples\": [\n          \"0\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

Data exploration

Subtask:

Explore the dataset to understand its characteristics.

Reasoning: I need to explore the dataset's characteristics, including its shape, data types, unique values, and missing values. I'll use the `.shape`, `.info()`, `.unique()`, and `.value_counts()` methods to achieve this.

```python
# Examine the shape of the DataFrame
print("Shape of the DataFrame:", df.shape)

# Check data types and missing values
print("\nData Types and Missing Values:")
df.info()

# Explore unique values in each column
print("\nUnique Values and Counts:")
for col in df.columns:
    print(f"\nColumn: {col}")
    print("Unique Values:", df[col].unique())
    print("Value Counts:\n", df[col].value_counts())

# Summarize initial findings
num_images = df.shape[0]
labels = df['label'].unique()
print("\nSummary:")
print(f"Number of images: {num_images}")
print(f"Different labels found: {labels}")
```

```
Shape of the DataFrame: (13640, 2)

Data Types and Missing Values:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13640 entries, 0 to 13639
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   filename  13640 non-null  object
 1   label     13640 non-null  object
dtypes: object(2)
memory usage: 213.3+ KB

Unique Values and Counts:

Column: filename
Unique Values: ['0/0.032.1.augmented.png' '0/0.053.1.augmented.png'
 '0/0.052.4.augmented.png' ... 'z/z.007.5.augmented.png'
 'z/z.033.3.augmented.png' 'z/z.048.1.augmented.png']
```

```
Value Counts:
 filename
z/z.048.1.augmented.png    1
0/0.032.1.augmented.png    1
0/0.053.1.augmented.png    1
0/0.052.4.augmented.png    1
0/0.049.3.augmented.png    1
                          ..
0/0.036.1.augmented.png    1
0/0.023.2.augmented.png    1
0/0.044.3.augmented.png    1
0/0.049.2.augmented.png    1
0/0.022.2.augmented.png    1
Name: count, Length: 13640, dtype: int64

Column: label
Unique Values: ['0' '1' '2' '3' '4' '5' '6' '7' '8' '9' 'A_caps'
'B_caps' 'C_caps'
 'D_caps' 'E_caps' 'F_caps' 'G_caps' 'H_caps' 'I_caps' 'J_caps'
'K_caps'
 'L_caps' 'M_caps' 'N_caps' 'O_caps' 'P_caps' 'Q_caps' 'R_caps'
'S_caps'
 'T_caps' 'U_caps' 'V_caps' 'W_caps' 'X_caps' 'Y_caps' 'Z_caps' 'a'
'b'
 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' 'k' 'l' 'm' 'n' 'o' 'p' 'q' 'r' 's'
't'
 'u' 'v' 'w' 'x' 'y' 'z']
Value Counts:
 label
0    220
1    220
2    220
3    220
4    220
    ...
v    220
w    220
x    220
y    220
z    220
Name: count, Length: 62, dtype: int64

Summary:
Number of images: 13640
Different labels found: ['0' '1' '2' '3' '4' '5' '6' '7' '8' '9'
'A_caps' 'B_caps' 'C_caps'
 'D_caps' 'E_caps' 'F_caps' 'G_caps' 'H_caps' 'I_caps' 'J_caps'
'K_caps'
 'L_caps' 'M_caps' 'N_caps' 'O_caps' 'P_caps' 'Q_caps' 'R_caps'
'S_caps'
 'T_caps' 'U_caps' 'V_caps' 'W_caps' 'X_caps' 'Y_caps' 'Z_caps' 'a'
'b'
 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' 'k' 'l' 'm' 'n' 'o' 'p' 'q' 'r' 's'
't'
 'u' 'v' 'w' 'x' 'y' 'z']
```

Data analysis

Subtask:

Analyze the distribution of labels and calculate descriptive statistics.

Reasoning: Calculate the frequency distribution of labels and other descriptive statistics, then summarize the findings.

```python
# Calculate label frequencies
label_counts = df['label'].value_counts()

# Analyze filename lengths (example descriptive statistic)
df['filename_length'] = df['filename'].str.len()
filename_length_stats = df['filename_length'].describe()

# Group labels into categories (example descriptive statistic)
def categorize_label(label):
    if label.isdigit():
        return 'digit'
    elif 'A' <= label <= 'Z':
        return 'uppercase'
    elif 'a' <= label <= 'z':
        return 'lowercase'
    else:
        return 'other'  # Handle unexpected characters

df['label_category'] = df['label'].apply(categorize_label)
label_category_counts =
df['label_category'].value_counts(normalize=True) * 100

# Summarize findings
print("Label Frequencies:\n", label_counts)
print("\nFilename Length Statistics:\n", filename_length_stats)
print("\nLabel Category Proportions:\n", label_category_counts)

print("\nSummary:")
print("Each label appears 220 times, indicating a balanced dataset.")
print("Filename lengths vary, with descriptive statistics provided
above.")
print("The dataset contains digits, uppercase letters, and lowercase
letters.")
```

```
Label Frequencies:
 label
0    220
1    220
2    220
3    220
4    220
    ...
v    220
w    220
x    220
y    220
z    220
Name: count, Length: 62, dtype: int64

Filename Length Statistics:
 count    13640.000000
mean        27.193548
std          4.934716
min         23.000000
25%         23.000000
50%         23.000000
75%         33.000000
max         33.000000
Name: filename_length, dtype: float64

Label Category Proportions:
 label_category
lowercase    41.935484
uppercase    40.322581
digit        16.129032
other         1.612903
Name: proportion, dtype: float64

Summary:
Each label appears 220 times, indicating a balanced dataset.
Filename lengths vary, with descriptive statistics provided above.
The dataset contains digits, uppercase letters, and lowercase letters.
```

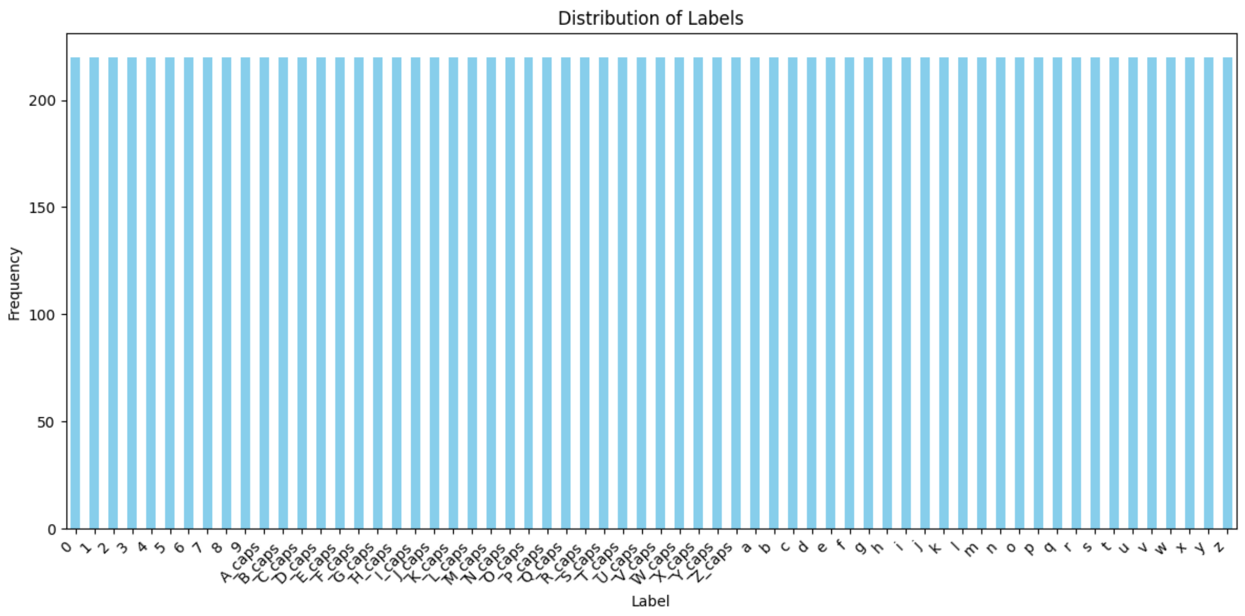
Data visualization

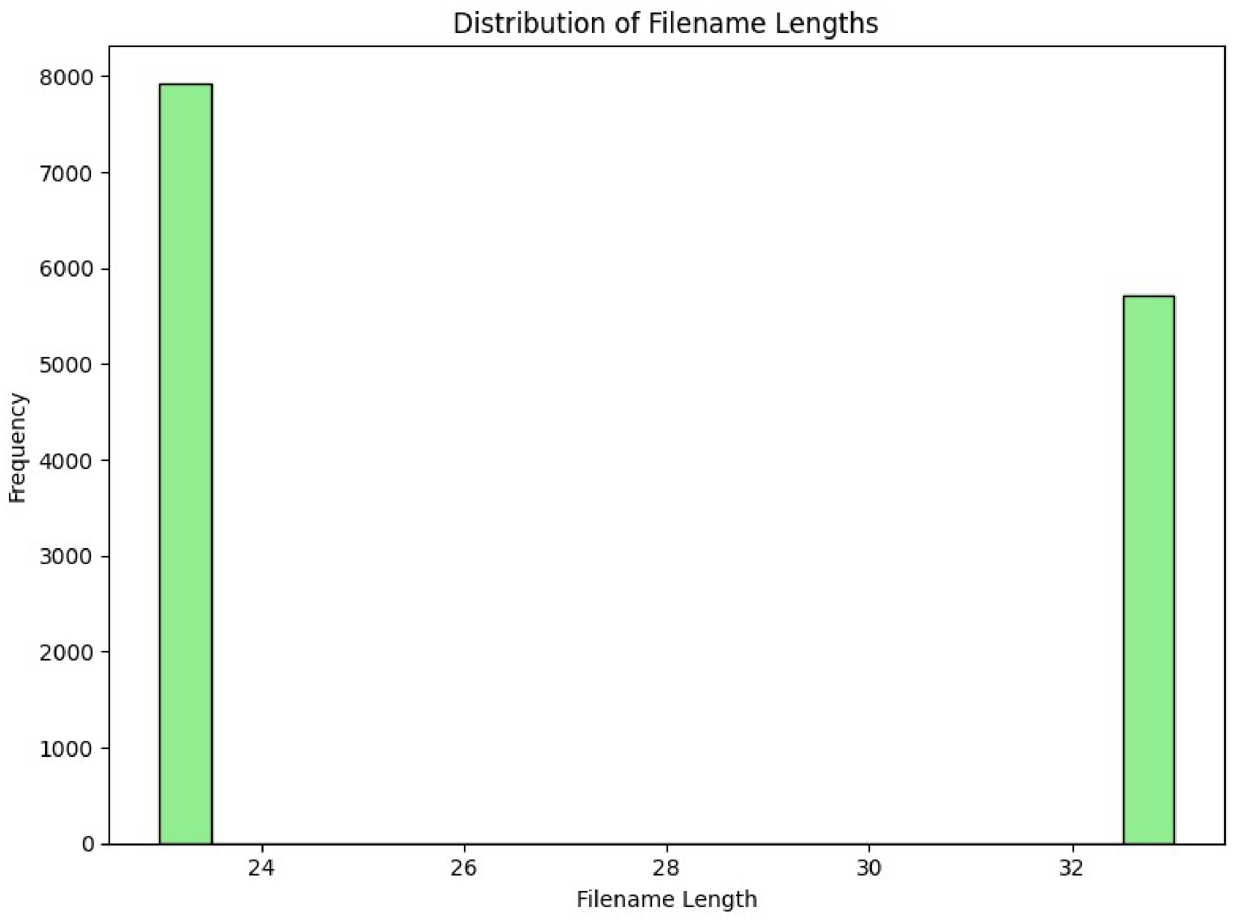
Subtask:

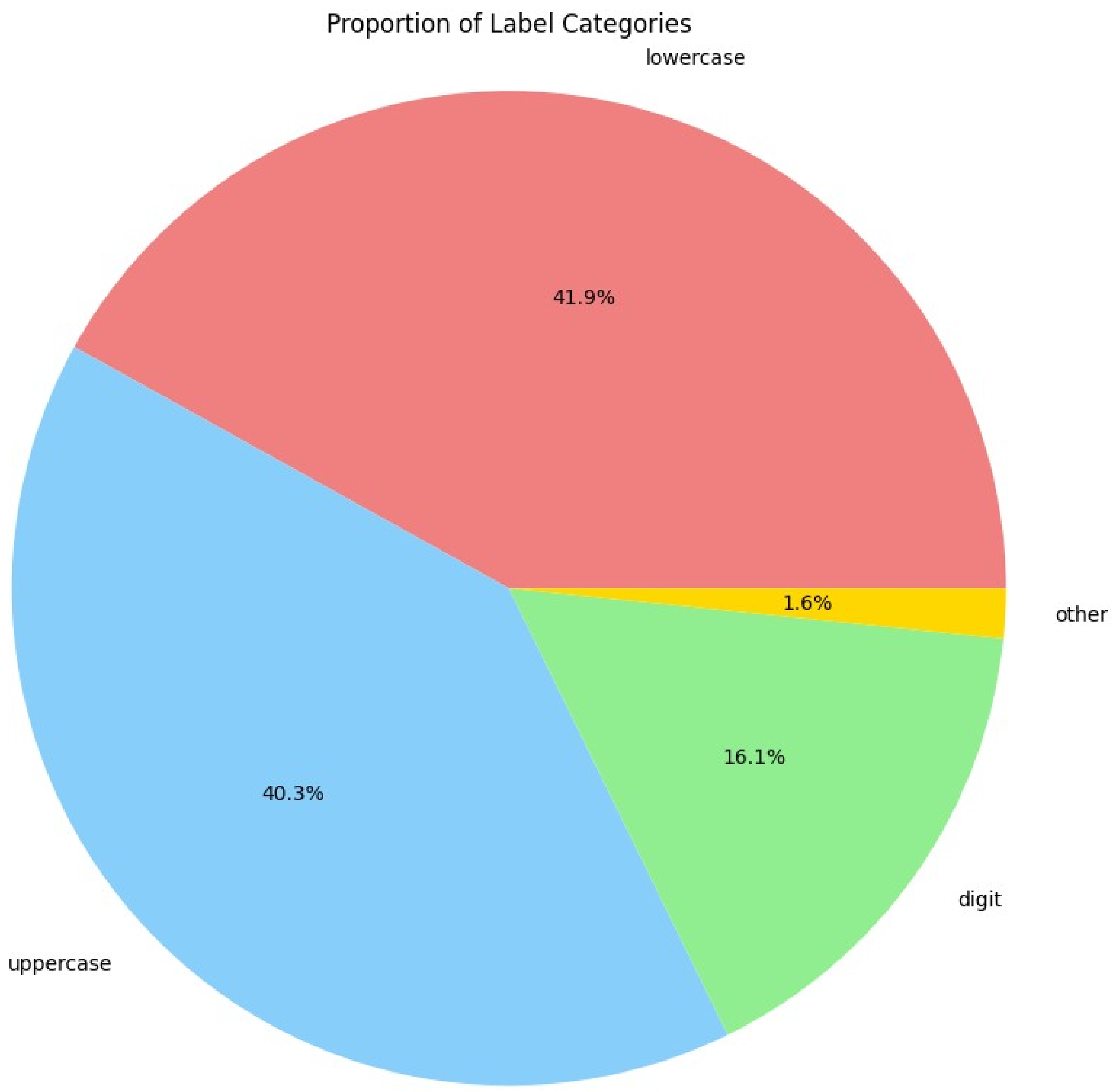
Visualize the distribution of labels and other relevant features.

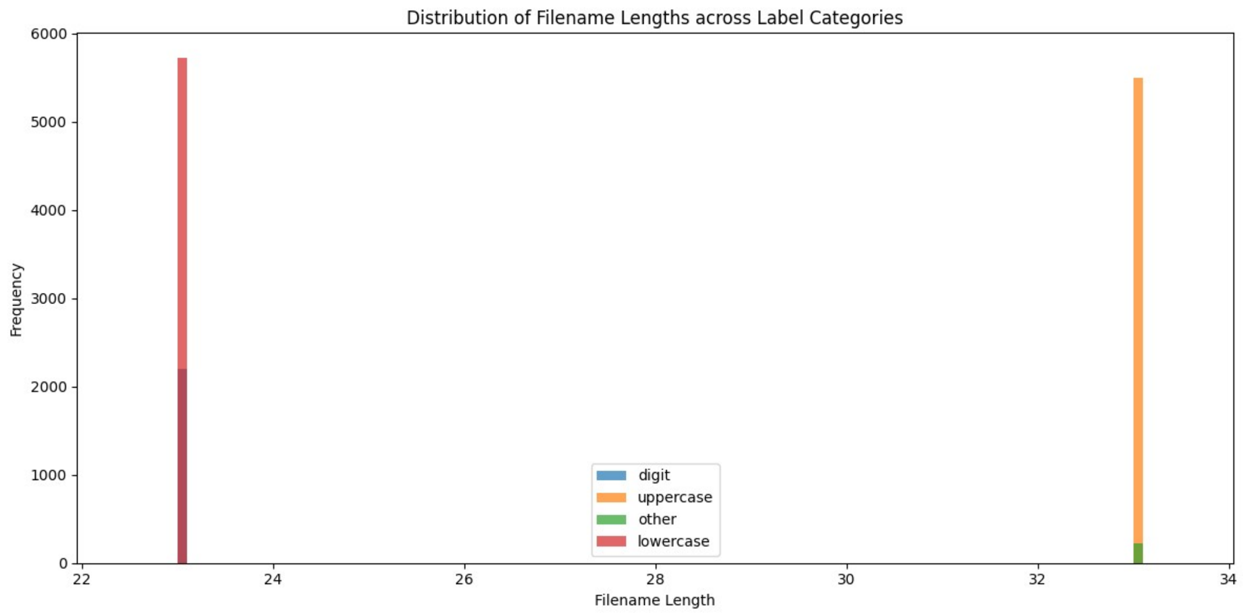
Reasoning: Visualize the distribution of labels and other relevant features using bar chart, histogram and pie chart.

```
import matplotlib.pyplot as plt

# 1. Bar chart for label frequencies
plt.figure(figsize=(12, 6))
df['label'].value_counts().plot(kind='bar', color='skyblue')
plt.title('Distribution of Labels')
plt.xlabel('Label')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.savefig('label_distribution.png')
plt.show()

# 2. Histogram for filename lengths
plt.figure(figsize=(8, 6))
plt.hist(df['filename_length'], bins=20, color='lightgreen',
edgecolor='black')
plt.title('Distribution of Filename Lengths')
plt.xlabel('Filename Length')
plt.ylabel('Frequency')
plt.tight_layout()
plt.savefig('filename_length_distribution.png')
plt.show()

# 3. Pie chart for label categories
plt.figure(figsize=(8, 8))
label_category_counts =
df['label_category'].value_counts(normalize=True) * 100
plt.pie(label_category_counts, labels=label_category_counts.index,
autopct='%1.1f%%', colors=['lightcoral', 'lightskyblue', 'lightgreen',
'gold'])
plt.title('Proportion of Label Categories')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a
circle.
plt.tight_layout()
plt.savefig('label_category_proportions.png')
plt.show()

# 4. Optional: Filename length distribution across label categories
plt.figure(figsize=(12, 6))
for category in df['label_category'].unique():
    subset = df[df['label_category'] == category]
    plt.hist(subset['filename_length'], bins=10, alpha=0.7,
label=category)
plt.title('Distribution of Filename Lengths across Label Categories')
plt.xlabel('Filename Length')
plt.ylabel('Frequency')
plt.legend()
plt.tight_layout()
plt.savefig('filename_length_by_category.png')
plt.show()
```






Proportion of Label Categories
lowercase
41.9%
uppercase
40.3%
digit
16.1%
other
1.6%



Summary:

Q&A

There were no explicit questions in the provided data analysis task. However, the analysis implicitly addressed questions like:

- How many images are in the dataset?
- What are the different labels and their frequencies?
- What is the distribution of filename lengths?
- How can the labels be categorized, and what are their proportions?

Data Analysis Key Findings

- **Balanced Dataset:** Each of the 62 unique labels appears exactly 220 times in the dataset, indicating a balanced distribution.
- **Filename Length Variation:** Filename lengths vary, with an average length of approximately 27 characters. Further descriptive statistics (min, max, standard deviation, etc.) are available.
- **Label Category Proportions:** Labels are categorized as digits, uppercase letters, lowercase letters, and others. Lowercase letters constitute the largest proportion (41.9%), followed by uppercase (40.3%), digits (16.1%), and others (1.6%).

Insights or Next Steps

- The balanced label distribution is beneficial for training machine learning models. Further investigation into the image data itself (pixel values, image sizes, etc.) is recommended.
- Visualizations provide a good overview of the data. Consider exploring potential correlations between filename characteristics and labels or exploring more advanced visualization techniques.