Combining different parts and outputing combined.csv

```
from google.colab import drive
import os
import pandas as pd
import numpy as np

# Mount the drive
drive.mount('/content/drive')

# Get the current working directory
cwd = os.path.abspath('/content/drive/MyDrive/CS 470 data')

# List all the files from the directory, filtering out directories
file_list = [f for f in os.listdir(cwd) if os.path.isfile(os.path.join(cwd, f))]

# Concatenate the DataFrames from each file, ignoring directories
df_concat = pd.concat([pd.read_csv(os.path.join(cwd, f)) for f in file_list], ignore_index=True)

# Remove columns named 'Unnamed: 0.1' and 'Unnamed: 0' first
df_concat = df_concat.drop(columns=['Unnamed: 0.1', 'Unnamed: 0'])

# Replace empty strings with NaN to uniformly represent missing values
df_concat.replace('', np.nan, inplace=True)
df_concat = df_concat.dropna(subset=df_concat.columns[1:], how='all')
df_concat.reset_index(drop=True, inplace=True)


df_concat.to_csv('/content/drive/MyDrive/CS 470 data/combined.csv', index=False)

df_concat
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
<ipython-input-2-7ec03f1359a4>:16: DtypeWarning: Columns (12) have mixed types. Specify dtype option on import or set low_memory=False.
  df_concat = pd.concat([pd.read_csv(os.path.join(cwd, f)) for f in file_list], ignore_index=True)
<ipython-input-2-7ec03f1359a4>:16: DtypeWarning: Columns (5,6) have mixed types. Specify dtype option on import or set low_memory=False.
  df_concat = pd.concat([pd.read_csv(os.path.join(cwd, f)) for f in file_list], ignore_index=True)
<ipython-input-2-7ec03f1359a4>:16: DtypeWarning: Columns (4,5,10) have mixed types. Specify dtype option on import or set low_memory=Fal
  df_concat = pd.concat([pd.read_csv(os.path.join(cwd, f)) for f in file_list], ignore_index=True)
```

	Major_School	Description	Date_Added	Status_Date	GRE_Stats	GPA	Degree	Citizenship	Attribute_1	Attribute_2	Attribute_
0	Education Policy, University Of Wisconsin-Madison	NaN	January 29, 2024	Rejected on 29 Jan	NaN	NaN	Masters	International	Fall 2024	NaN	Na
1	Computer Science, University Of Illinois	CV area. Did my undergrad at same school, One ...	January 29, 2024	Accepted on 29 Jan	NaN	GPA 4.00	PhD	International	Fall 2024	NaN	Na
2	Astronomy, Indiana University Bloomington	NaN	January 29, 2024	Rejected on 28 Jan	NaN	NaN	PhD	American	Fall 2024	NaN	Na
3	Physics, Penn State University	Accepted for high energy theory with a focus o...	January 29, 2024	Accepted on 29 Jan	NaN	GPA 3.00	PhD	International	Fall 2024	NaN	Na
4	Integrated Program In Biochemistry (IPiB), Uni...	NaN	January 29, 2024	Accepted on 29 Jan	NaN	NaN	PhD	American	Fall 2024	NaN	Na
...	...	...	...	...	...	...	...	...	...	...	..
22040845	Advertising, University of Texas, Austin (UT A...	NaN	February 11, 2006	Accepted on 2 Feb	NaN	NaN	NaN	NaN	NaN	NaN	Na

GPA and GRE score Only for US applicants (GPA difficult to normalize and convert between different countries)

```
# Filtering out 'international' from 'Citizenship' column, ensuring proper handling of NaN values
df_USONLY = df_concat[(~df_concat['Citizenship'].str.contains("international", case=False, na=False)) & df_concat['Citizenship'].notna()]
df_USONLY

# Make data frame with Major/School, GPA, GRE, and Application Cycle, clean unnecessary data
df_gpaandgre = df_USONLY[['Major_School', 'GPA', 'GRE_Stats', 'Attribute_1', 'Status_Date']].rename(columns={'Attribute_1': 'Application Cycl
df_gpaandgre = df_gpaandgre.dropna()
df_gpaandgre['GPA'] = df_gpaandgre['GPA'].str.replace('GPA ', '').astype(float)
df_gpaandgre = df_gpaandgre[df_gpaandgre['Application Cycle'] != 'Other']

# Simplify values to either Accept, Wailist, or Reject, remove other values
def simplify_status(status):
    if "accept" in status.lower():
        return "Accept"
    elif "wait listed" in status.lower():
        return "Waitlist"
    elif "interview" in status.lower():
        return None
    else:
        return "Reject"
df_gpaandgre['Status'] = df_gpaandgre['Status_Date'].apply(simplify_status)
df_gpaandgre = df_gpaandgre.dropna(subset=['Status'])

# Reset the index
df_gpaandgre.reset_index(drop=True, inplace=True)

# Export to File
df_gpaandgre.to_csv('/content/drive/MyDrive/CS 470 data/gpaandgre.csv', index=False)
```

```
<ipython-input-22-78538c01cd12>:21: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  df_gpaandgre['Status'] = df_gpaandgre['Status_Date'].apply(simplify_status)
```

```
df_gpaandgre
```

	Major_School	GPA	GRE_Stats	Application Cycle	Status_Date	Status
0	Human Ecology PhD - Design Studies, University...	2.18	GRE 157; GRE V 155; GRE AW 4.00	Fall 2024	Accepted on 24 Jan	Accept
1	Political Science, Brown University	3.80	GRE 155; GRE V 168	Fall 2024	Rejected on 29 Jan	Reject
2	Political Science, Brown University	4.00	GRE 158; GRE V 165; GRE AW 4.50	Fall 2024	Rejected on 29 Jan	Reject
3	Sociology, Emory University	3.46	GRE 161; GRE V 163; GRE AW 3.00	Fall 2024	Rejected on 29 Jan	Reject
4	Sociology, Indiana University Bloomington	3.46	GRE 163; GRE V 161; GRE AW 3.00	Fall 2024	Rejected on 26 Jan	Reject
...	...	...	...	...	...	...
	Communication		GRE 570; GRE			

Most popular Major_School

```
major_school_counts = df_gpaandgre['Major_School'].value_counts().reset_index(name='Count')

major_school_counts.rename(columns={'index': 'Major_School'}, inplace=True)

major_school_counts.head()
```

	Major_School	Count
0	Speech Language Pathology, University Of South...	910
1	Economics, University Of Michigan (Ann Arbor)	884
2	Economics, Stanford University	858
3	Economics, Yale University	754
4	Economics, University Of Chicago	624

Curious about Chemistry

```
# Filter the DataFrame for programs in Chemistry
chemistry_programs = df_gpaandgre[df_gpaandgre['Major_School'].str.contains("Chemistry", case=False)]

# Extract and standardize university names from the Major_School entries for easier comparison
chemistry_programs['University'] = chemistry_programs['Major_School'].apply(lambda x: x.split(',')[1] if ',' in x else x)

duplicate_chemistry_programs = chemistry_programs[chemistry_programs.duplicated(subset=['University'], keep=False)]
# Reset the index
duplicate_chemistry_programs.reset_index(drop=True, inplace=True)
# Export to File
duplicate_chemistry_programs.to_csv('/content/drive/MyDrive/CS 470 data/duplicate_chemistry_programs.csv', index=False)
duplicate_chemistry_programs
```

```
<ipython-input-39-876d250a22a5>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
  chemistry_programs['University'] = chemistry_programs['Major_School'].apply(lambda x:
```

	Major_School	GPA	GRE_Stats	Application Cycle	Status_Date	Status	University
0	Chemistry, Harvard University	4.00	GRE 162; GRE V 161; GRE AW 4.50	Fall 2024	Accepted on 23 Jan	Accept	Harvard University
1	Chemistry, University Of California, Berkeley	4.00	GRE 162; GRE V 161; GRE AW 4.50	Fall 2024	Accepted on 16 Jan	Accept	University Of California
2	Chemistry, University Of Chicago	4.00	GRE 161; GRE V 162; GRE AW 4.50	Fall 2024	Accepted on 5 Jan	Accept	University Of Chicago
3	Chemistry, Brown University	3.92	GRE 164; GRE V 165; GRE AW 4.50	Fall 2023	Rejected on 3 Feb	Reject	Brown University
			GRE 164;				

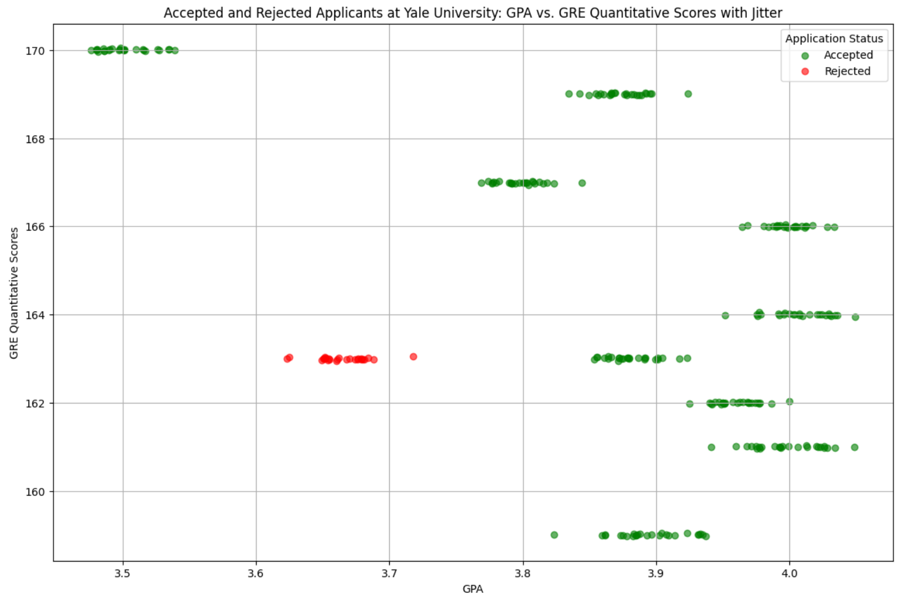
```
duplicate_chemistry_programs_count = duplicate_chemistry_programs['University'].value_counts().reset_index(name='Count')
duplicate_chemistry_programs_count.head()
```

	index	Count
0	University Of California	416
1	Yale University	260
2	Stanford University	234
3	University Of Chicago	208
4	Northwestern University	156

GPA and GRE rates for Yale University (this part is bugged, pls fix later)

```
import matplotlib.pyplot as plt
import numpy as np
# Filter the dataset for Yale University applicants only, regardless of application cycle
yale_university_applicants = duplicate_chemistry_programs[duplicate_chemistry_programs['University'].str.contains("Yale")]

# Display the DataFrame for Yale University applicants
yale_university_applicants.reset_index(drop=True, inplace=True)
yale_university_applicants.to_csv('/content/drive/MyDrive/CS 470 data/yale_university_applicants.csv', index=False)
import matplotlib.pyplot as plt
yale_university_applicants['GRE_Stats'] = yale_university_applicants['GRE_Stats'].astype(str)
yale_university_applicants['GRE_Stats'] = yale_university_applicants['GRE_Stats'].str.extract(r'GRE (\d+)').astype(float)
yale_university_applicants

# fall_2021_applicants = yale_university_applicants[yale_university_applicants['Application Cycle'] == 'Fall 2021']
# fall_2021_applicants
selected_columns = yale_university_applicants[['GPA', 'GRE_Stats', 'Application Cycle', 'Status']]
selected_columns.reset_index(drop=True, inplace=True)
selected_columns

gpa = selected_columns['GPA'].values
statuses = selected_columns['Status'].values
gre_quantitative = selected_columns['GRE_Stats'].values

accepted_indices = np.where(selected_columns['Status'] == 'Accept')
rejected_indices = np.where(selected_columns['Status'] == 'Reject')

gpa_accepted = gpa[accepted_indices]
gre_quantitative_accepted = gre_quantitative[accepted_indices]

gpa_rejected = gpa[rejected_indices]
gre_quantitative_rejected = gre_quantitative[rejected_indices]

# Introduce jitter to the data for better visualization
jitter = 0.02  # Adjust this value as needed for the desired amount of jitter

gpa_accepted_jittered = gpa_accepted + np.random.normal(0, jitter, size=len(gpa_accepted))
gre_accepted_jittered = gre_quantitative_accepted + np.random.normal(0, jitter, size=len(gre_quantitative_accepted))

gpa_rejected_jittered = gpa_rejected + np.random.normal(0, jitter, size=len(gpa_rejected))
gre_rejected_jittered = gre_quantitative_rejected + np.random.normal(0, jitter, size=len(gre_quantitative_rejected))

# Prepare the plot
plt.figure(figsize=(14, 9))

# Plot accepted applicants with green points and jitter
plt.scatter(gpa_accepted_jittered, gre_accepted_jittered, color='green', label='Accepted', alpha=0.6)

# Plot rejected applicants with red points and jitter
plt.scatter(gpa_rejected_jittered, gre_rejected_jittered, color='red', label='Rejected', alpha=0.6)

# Adding plot title and labels
plt.title('Accepted and Rejected Applicants at Yale University: GPA vs. GRE Quantitative Scores with Jitter')
plt.xlabel('GPA')
plt.ylabel('GRE Quantitative Scores')
plt.legend(title='Application Status')
plt.grid(True)

# Display the plot
plt.show()
```

```
<ipython-input-139-be4906f1526e>:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
  yale_university_applicants['GRE_Stats'] = yale_university_applicants['GRE_Stats'].asty
<ipython-input-139-be4906f1526e>:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
  yale_university_applicants['GRE_Stats'] = yale_university_applicants['GRE_Stats'].str.
```


Accepted and Rejected Applicants at Yale University: GPA vs. GRE Quantitative Scores with Jitter
Application Status
Accepted
Rejected
GRE Quantitative Scores
GPA