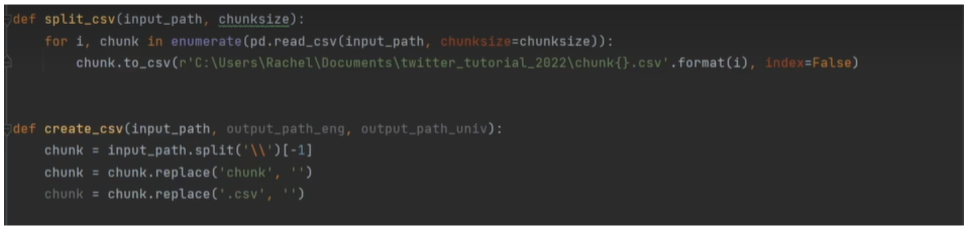
11.Selecting Data

We Want to split our csv files into the appropriate categories that being English and universal. We will do more splits for various categories later but these are the main ones we will be targeting now. We will be taking those split sections and creating new csv as shown below.

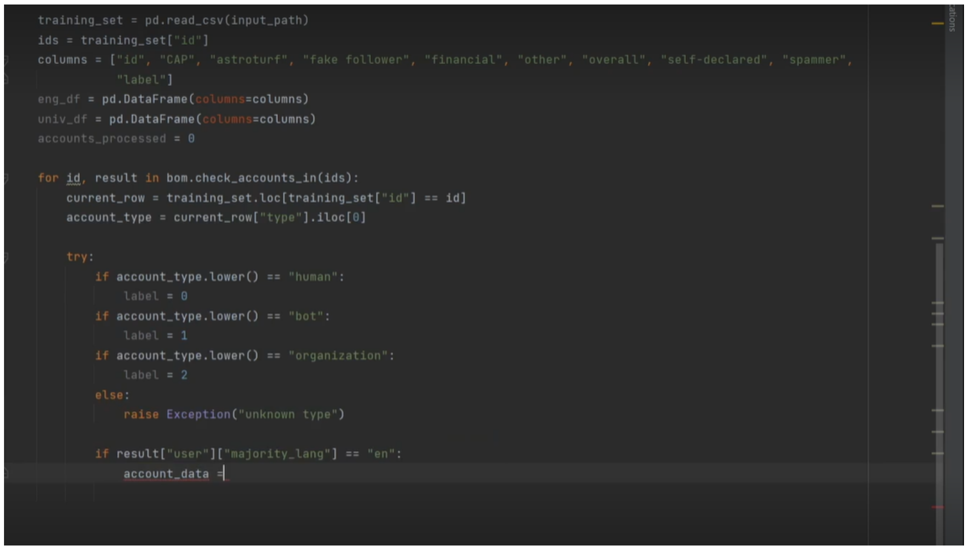
The code looks like this:



```
def split_csv(input_path, chunksize):
    for i, chunk in enumerate(pd.read_csv(input_path, chunksize=chunksize)):
        chunk.to_csv(r'C:\Users\Rachel\Documents\twitter_tutorial_2022\chunk{}.csv'.format(i), index=False)


def create_csv(input_path, output_path_eng, output_path_univ):
    chunk = input_path.split('\\')[-1]
    chunk = chunk.replace('chunk', '')
    chunk = chunk.replace('.csv', '')
```

We are then going to checks all the ids and label if each account is either a human, bot, or organization and we assign a numeric value for each of those categories for future utilization.



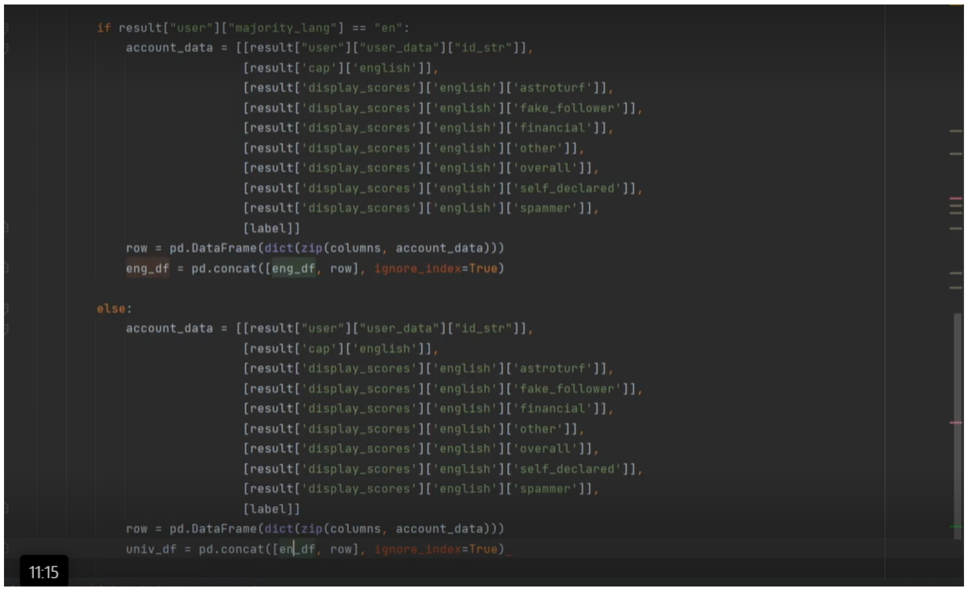
```
    training_set = pd.read_csv(input_path)
    ids = training_set["id"]
    columns = ["id", "CAP", "astroturf", "fake follower", "financial", "other", "overall", "self-declared", "spammer",
               "label"]
    eng_df = pd.DataFrame(columns=columns)
    univ_df = pd.DataFrame(columns=columns)
    accounts_processed = 0

    for id, result in bom.check_accounts_in(ids):
        current_row = training_set.loc[training_set["id"] == id]
        account_type = current_row["type"].iloc[0]

        try:
            if account_type.lower() == "human":
                label = 0
            if account_type.lower() == "bot":
                label = 1
            if account_type.lower() == "organization":
                label = 2
            else:
                raise Exception("unknown type")

            if result["user"]["majority_lang"] == "en":
                account_data =
```

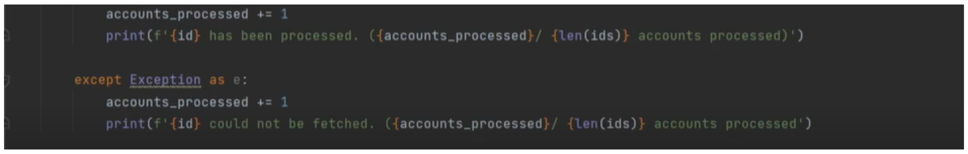
We want to get the account data for english as well as universal. We also want to create a data frame for both english and universal so we use an if else statement like this while utilizing pandas:



```
if result["user"]["majority_lang"] == "en":
    account_data = [[result["user"]["user_data"]["id_str"]],
                    [result['cap']['english']],
                    [result['display_scores']['english']['astroturf']],
                    [result['display_scores']['english']['fake_follower']],
                    [result['display_scores']['english']['financial']],
                    [result['display_scores']['english']['other']],
                    [result['display_scores']['english']['overall']],
                    [result['display_scores']['english']['self_declared']],
                    [result['display_scores']['english']['spammer']],
                    [label]]
    row = pd.DataFrame(dict(zip(columns, account_data)))
    eng_df = pd.concat([eng_df, row], ignore_index=True)

else:
    account_data = [[result["user"]["user_data"]["id_str"]],
                    [result['cap']['english']],
                    [result['display_scores']['english']['astroturf']],
                    [result['display_scores']['english']['fake_follower']],
                    [result['display_scores']['english']['financial']],
                    [result['display_scores']['english']['other']],
                    [result['display_scores']['english']['overall']],
                    [result['display_scores']['english']['self_declared']],
                    [result['display_scores']['english']['spammer']],
                    [label]]
    row = pd.DataFrame(dict(zip(columns, account_data)))
    univ_df = pd.concat([en_df, row], ignore_index=True)
```

We want to also print out the result and other important information to the console to see if our program is working. We do this by adding a print statement under the code above. There is no guarantee that botometer will be able to access every twitter account so we add an exception statement like this:

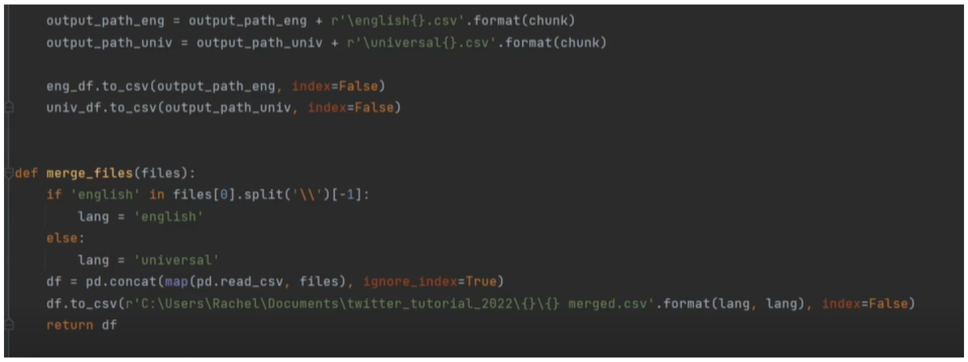


```
        accounts_processed += 1
        print(f'{id} has been processed. ({accounts_processed}/ {len(ids)} accounts processed)')

    except Exception as e:
        accounts_processed += 1
        print(f'{id} could not be fetched. ({accounts_processed}/ {len(ids)} accounts processed')
```

Which will print in the console the twitter id of any account that could not be fetched by the botometer API.

We then update our paths that we will use to merge the files and turn the dfs to the csv's as shown below we will split them by language as before it shoud look like this:

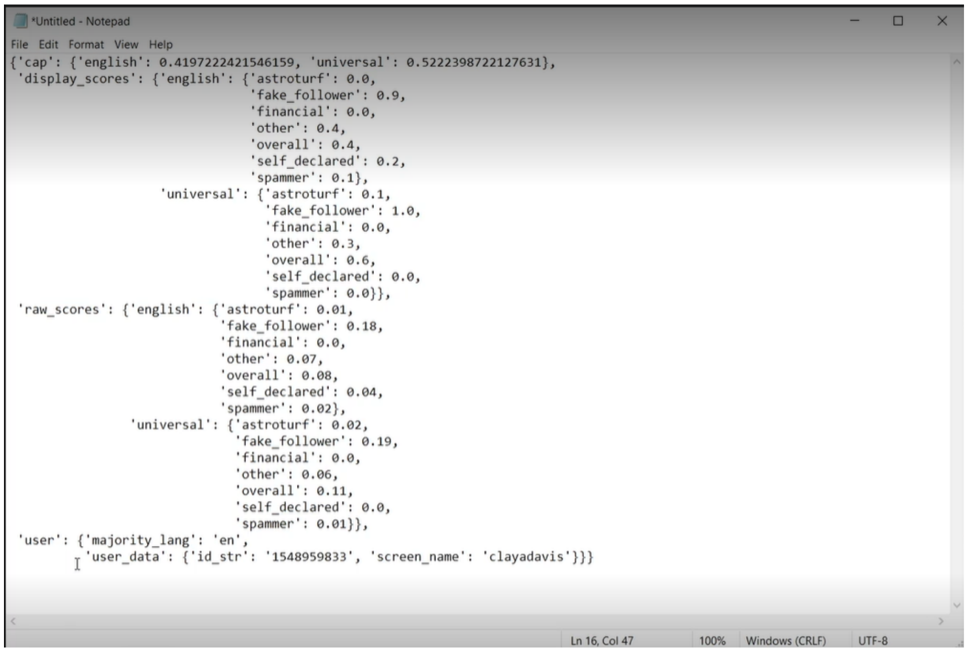


```
    output_path_eng = output_path_eng + r'\english{}.csv'.format(chunk)
    output_path_univ = output_path_univ + r'\universal{}.csv'.format(chunk)

    eng_df.to_csv(output_path_eng, index=False)
    univ_df.to_csv(output_path_univ, index=False)


def merge_files(files):
    if 'english' in files[0].split('\\')[-1]:
        lang = 'english'
    else:
        lang = 'universal'
    df = pd.concat(map(pd.read_csv, files), ignore_index=True)
    df.to_csv(r'C:\Users\Rachel\Documents\twitter_tutorial_2022\{}\{} merged.csv'.format(lang, lang), index=False)
    return df
```

Run your code and after the twitter accounts are tabulated in the left side of your pycharms application you should see a csv file appear. When you open the csv file in the text editor it should look similar to this (do this by right clicking the file and going to “Open With” then click the notepad or any text editor you are using:



```
{'cap': {'english': 0.4197222421546159, 'universal': 0.5222398722127631},
 'display_scores': {'english': {'astroturf': 0.0,
                                'fake_follower': 0.9,
                                'financial': 0.0,
                                'other': 0.4,
                                'overall': 0.4,
                                'self_declared': 0.2,
                                'spammer': 0.1},
                    'universal': {'astroturf': 0.1,
                                  'fake_follower': 1.0,
                                  'financial': 0.0,
                                  'other': 0.3,
                                  'overall': 0.6,
                                  'self_declared': 0.0,
                                  'spammer': 0.0}},
 'raw_scores': {'english': {'astroturf': 0.01,
                            'fake_follower': 0.18,
                            'financial': 0.0,
                            'other': 0.07,
                            'overall': 0.08,
                            'self_declared': 0.04,
                            'spammer': 0.02},
                'universal': {'astroturf': 0.02,
                              'fake_follower': 0.19,
                              'financial': 0.0,
                              'other': 0.06,
                              'overall': 0.11,
                              'self_declared': 0.0,
                              'spammer': 0.01}},
 'user': {'majority_lang': 'en',
          'user_data': {'id_str': '1548959833', 'screen_name': 'clayadavis'}}}
```

OVERALL: This part (part 11 – 12) of the document does not match the tutorial.

Note: The font and point are also inconsistent.

Part 13 is also missing.