

fx-update_prediction_data

Link

<https://github.com/reddit-archive/reddit/blob/753b17407e9a9dca09558526805922de24133d53/r2/r2/lib/inventory.py>

▼ Code

```
def update_prediction_data():
    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)

    # combine front page values (sometimes frontpage gets '' for its name)
    if '' in min_daily_by_sr:
        fp = DefaultSR.name.lower()
        min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr['']
        del min_daily_by_sr['']

    filtered = {sr_name: num for sr_name, num in min_daily_by_sr.iteritems()
                 if num > 100}
    PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)
```

▼ Documentation

```
"""Fetch prediction data and write it to cassandra."""
```

Prompt

As a code documentation assistant, you are programmed to document at the function level of the given code snippet. Your approach involves placing comments directly under the def statement of the function. The output should be the entire code along with the documentation of the function written. Code is as follows:

```
"""
def update_prediction_data():
    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)

    if '' in min_daily_by_sr:
        fp = DefaultSR.name.lower()
        min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr['']
        del min_daily_by_sr['']
```

```

        filtered = {sr_name: num for sr_name, num in min_daily_by_sr.iteritems()
                     if num > 100}
        PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)
    """

```

Responses

ChatGPT

```

def update_prediction_data():
    """
    Update prediction data based on minimum daily pageviews by subreddit.

    This function calculates the minimum daily pageviews by subreddit and updates the prediction data with the filtered data where the pageviews are greater than 100.

    Parameters:
        None

    Returns:
        None

    Dependencies:
        - _min_daily_pageviews_by_sr: A private function used to fetch minimum daily pageviews by subreddit.
        - DefaultSR: A constant representing the default subreddit.
        - NDAYS_TO_QUERY: The number of days to query for pageviews.
        - MIN_DAILY_CASS_KEY: The key for storing the minimum daily pageviews data in PromoMetrics.

    Note:
        - If a subreddit name is empty (''), it is considered as the default subreddit.
        - Pageviews below 100 are not included in the filtered data.

    Example:
        update_prediction_data()
    """
    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)

    if '' in min_daily_by_sr:
        fp = DefaultSR.name.lower()
        min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr['']
        del min_daily_by_sr['']

    filtered = {sr_name: num for sr_name, num in min_daily_by_sr.items()
                 if num > 100}
    PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)

```

GPT4

```
def update_prediction_data():
    """
    Updates the prediction data for daily pageviews across subreddits.

    The function queries data for a predefined number of days (NDAYS_TO_QUERY)
    to find the minimum daily pageviews by subreddit. It then performs two key operations:
    1. If the 'default subreddit' (denoted by '') is present in the data, it merges
       its pageviews with those of the Default Subreddit, and then removes the
       default entry.
    2. Filters out subreddits that have daily pageviews less than a specified threshold
       (in this case, 100).

    It finally updates the PromoMetrics with the filtered data, storing it under
    the key MIN_DAILY_CASS_KEY.

    Internals:
    - `_min_daily_pageviews_by_sr(NDAYS_TO_QUERY)` : Internal function call to retrieve
      minimum daily pageviews by subreddit.
    - `PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)` : Saves the filtered data in
      PromoMetrics under the specified key.

    Note: This function does not return any value. It is used for updating internal
    data structures only.
    """

    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)

    if '' in min_daily_by_sr:
        fp = DefaultSR.name.lower()
        min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr['']
        del min_daily_by_sr['']

    filtered = {sr_name: num for sr_name, num in min_daily_by_sr.iteritems()
                 if num > 100}
    PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)
```

Bard

```
def update_prediction_data():
    """Updates the prediction data for the given set of SRs.

    Args:
        sr_names: A list of SR names to update prediction data for.

    Returns:
        None
    """

    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)
```

```

if '' in min_daily_by_sr:
    fp = DefaultSR.name.lower()
    min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr['']
    del min_daily_by_sr['']

filtered = {sr_name: num for sr_name, num in min_daily_by_sr.iteritems()
            if num > 100}
PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)

```

LLama2

```

def update_prediction_data():
    """
    Updates the prediction data by calculating the minimum daily page views for each SR and

    :param NDAYS_TO_QUERY: The number of days to query for page view data
    :type NDAYS_TO_QUERY: int

    :return: None
    """
    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)

    if '' in min_daily_by_sr:
        fp = DefaultSR.name.lower()
        min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr['']
        del min_daily_by_sr['']

    filtered = {sr_name: num for sr_name, num in min_daily_by_sr.iteritems() if num > 100}
    PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)

```

StarChat

```

def update_prediction_data():
    """This function updates the prediction data stored in Cassandra by
    querying the top N search results from Google Search Console for each day
    over the last week."""
    # Query Google Search Console for daily minimum page views by device type
    min_daily_by_sr = _min_daily_pageviews_by_sr(NDAYS_TO_QUERY)

    # Handle case where an empty string was returned (likely due to a bug)
    if "" in min_daily_by_sr:
        fp = DefaultSR.name.lower()
        min_daily_by_sr[fp] = min_daily_by_sr.get(fp, 0) + min_daily_by_sr[""]
        del min_daily_by_sr[""]

    # Filter out any search result that didn't meet our threshold
    filtered = {sr_name: num for sr_name, num in min_daily_by_sr.items()
                if num > MIN_PAGEVIEWS_THRESHOLD}

    # Update Cassandra table with filtered data

```

```
PromoMetrics.set(MIN_DAILY_CASS_KEY, filtered)
```

```
ddddddcccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```
