**Popularity**

Assuming fresh views have the same weight as old views.

For non-interactive websites (no like, share, comment), then factors regarding like, share and comment are not counted.

Popularity Score=*w1*×Views+*w2*×Unique Visitors+*w*3×Normalized Dwell Time

where:

* Views: Total number of page loads for the article.
* Unique Visitors: Number of distinct users.
* Normalized Dwell Time: Average dwell time per 100 words or equi. Normalized Dwell Time (per 100 words)=Avg. Dwell Time (seconds)/Article length (words) ×100

As a start, we adopt the following weight values and adjust later.

Score=1×Views+5×Unique Visitors+10×Normalized Dwell Time

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Trending**

**Use recency-based for all matrices, formula same as Popularity Score**

Fresh views should be **weighted more heavily** than older views in the popularity score, reflecting how new engagement indicates current relevance and reach. The standard method is to apply a **decay function**—most commonly exponential decay—to reduce the impact of older views, ensuring the score favors recent activity while still accounting for historic performance.

Score=*w1*×× Wt Views+*w2*×Recent Unique Visitors+*w3*×Recent Normalized Dwell Time

A widely-used formula is:

Weighted View=

*N*

∑

*i=1 Vi*×*e*−*λ*(*T*now−*Ti*)

Where:

• Vi is number of views at time Ti

• Tnow is the current time

• *λ* Is the decay rate (a constant that sets how quickly older views lose influence; typically between 0.1 and 0.5 for daily/weekly scaling)

• *e*−*λ*(Tnow - Ti) is the decay factor where e = 2.71828

**Setting the Weight (Decay Rate)**

• For **news**, choose a higher decay rate ( *λ* approx 0.3-0.5 ) to make the score sensitive to current events and trends, so views from a few days ago have significantly less influence.

• For **evergreen articles**, use a lower rate ( *λ* approx 0.1-0.3 ), balancing lasting value and recent relevance.

**Practical Example**

Suppose views per day for the past week: [Mon=50, Tue=40, Wed=30, Thu=20, Fri=10] and today is Saturday.  
With *λ* = 0.5 and “age” in days:

• Sat: 1 day ago, decay = e-0.5 \* 1 approx 0.61

• Fri: 2 days ago, decay = e-0.5 \* 2 approx 0.37

• ... and so on

Each day’s views are multiplied by its decay factor, summed for total weighted views.

**Usage in Popularity Score**

Replace raw “Views” in your main formula with “Weighted Views”:

This gives the score strong recency awareness, helping track both **recent reach and overall engagement**.

**Summary**

• Use **exponential decay** to down-weight older views

• For news: *λ* = 0.3-0.5 is recommended for starting

• Adjust over time to fit site behavior and content goals

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Spec\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Backend Process to calculate score:

At regular interval, run batch job to:

**Derive Popular Score**

StartProcessDT = latest lastprocessDT from PopScore file. If null, then assign it with date time before the oldest user activity date time.

ProcessDT = current system DT

Scan user activity.processdt > StartProcessDT and user activity.processdt <= processDT to determine active content ID list for processing.

For content id in list:

Get article.wordcount

PopScore.Views = User activity record count for same content ID

PopScore.unique visitor = distinct session ID count for same content ID

PopScore.ttlDwelltime (sec.) = summation total of all dwelltime for same content ID

Get PopScore record with the same content ID. If record is found, then fetch PrevViews, PrevUniqueVisitor, PrevTtlDwellTime else Prev\* = 0

Views = PopScore.Views + PrevViews

UniqueVisitorCount = PopScore.UniqueVisitor + PrevUniqueVisitor

ttlDwellTime =PopScorfe.ttlDwellTime + PrevDwellTime

Normalized Dwell Time (per 100 words)=Avg. Dwell Time (seconds)/Article word count (words) ×100 where Avg Dwell Time = TtlDwellTime/Unique Visitor

Calculate popScore

*= w1*×Views+*w2*×Unique Visitor+*w*3×Normalized Dwell Time

LastViewDate = User Activity last view date

If content ID record does not exist in PopScore, add record, assign FirstViewDate = first ViewDate of article in user activity else

update PopScore file’s all fields except FirstViewDate, Prev\*,

**Derive TrendScore**

Clear all TrendScore records in system

Define ElapseDay (no. of days considered in trending analysis, eg. 30 days)

e=2.71828

Ti = current date - ElapseDay

Tnow = current date

ProcessDT = Current System DT

Fetch all user activity with view date greater than or equal to Ti and user activity view date time <= ProcessDTto determine *content ID* list

Looping content ID list:

For each content ID,

Get Article.articletype (N or E).

Determine *λ* to use from articletype. If articletype = E then *λ = 0.3 else*  *λ = 0.5*

Get Article.wordcount

Loop user activity from Ti to Tnow,

For each distinct date i , determine the total views Vi and age day (day from view date to Tnow) for i, calculate Weighted views for day i : *Vi*×*e*−*λ*(*T*now−*Ti*)

WeightedView of this contentID = Sum of all weighted views per day from i to current date, now

Get the content ID’s recent unique visitors, recent normalized dwell time from start view date (first view date>=Ti) to current date in user activity.

Calculate TrendScore where

*w1*×WtViews+*w2*×Recent Unique Visitors+*w*3×Recent Normalized Dwell Time

LastViewDate = latest UserActivity view date in the loop

Add record to TrendScore file

End of content ID

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*File Layout\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PopScore

|  |
| --- |
| Key:contentID |
| FirstViewDate |
| LastViewDate |
| PopScore |
| Views |
| DistinctVisitor |
| TtlDwellTime |
| NormalizedDwellTime |
| ProcessDT |

TrendScore

|  |
| --- |
| Key:contentID |
| StartViewDate |
| LastViewDate |
| TrendScore |
| Views |
| DistinctVisitor |
| TtlDwellTime |
| NormalizedDwellTime |
| ProcessDT |