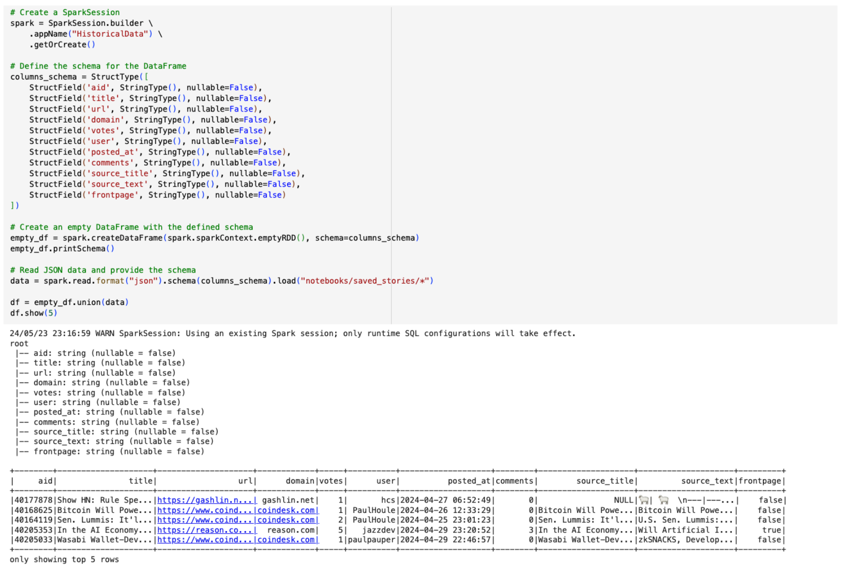
3 Assignment 3: Streaming Analytics on Text data

3.1 Data Collection

In this assignment, we construct a predictive model on Spark streaming by using the text data from the website ”hacker news”. We set the trigger time as 30 seconds, to make sure the data collection and lower latency. After collecting the text on the streaming for ten days, 4887 news articles are collected. 4698 news are finally taken into account after checking the duplication. The columns as below are set as the schema at first for the dataframe to read the one-piece json data.

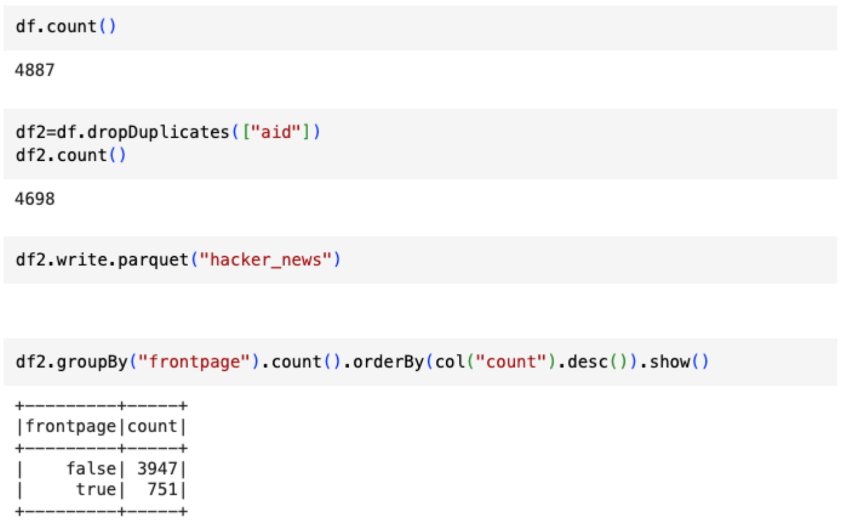


```python
# Create a SparkSession
spark = SparkSession.builder \
    .appName("HistoricalData") \
    .getOrCreate()

# Define the schema for the DataFrame
columns_schema = StructType([
    StructField('aid', StringType(), nullable=False),
    StructField('title', StringType(), nullable=False),
    StructField('url', StringType(), nullable=False),
    StructField('domain', StringType(), nullable=False),
    StructField('votes', StringType(), nullable=False),
    StructField('user', StringType(), nullable=False),
    StructField('posted_at', StringType(), nullable=False),
    StructField('comments', StringType(), nullable=False),
    StructField('source_title', StringType(), nullable=False),
    StructField('source_text', StringType(), nullable=False),
    StructField('frontpage', StringType(), nullable=False)
])

# Create an empty DataFrame with the defined schema
empty_df = spark.createDataFrame(spark.sparkContext.emptyRDD(), schema=columns_schema)
empty_df.printSchema()

# Read JSON data and provide the schema
data = spark.read.format("json").schema(columns_schema).load("notebooks/saved_stories/*")

df = empty_df.union(data)
df.show(5)
```

```
24/05/23 23:16:59 WARN SparkSession: Using an existing Spark session; only runtime SQL configurations will take effect.
root
 |-- aid: string (nullable = false)
 |-- title: string (nullable = false)
 |-- url: string (nullable = false)
 |-- domain: string (nullable = false)
 |-- votes: string (nullable = false)
 |-- user: string (nullable = false)
 |-- posted_at: string (nullable = false)
 |-- comments: string (nullable = false)
 |-- source_title: string (nullable = false)
 |-- source_text: string (nullable = false)
 |-- frontpage: string (nullable = false)

+--------+--------------------+--------------------+------------+-----+----------+-------------------+--------+--------------------+--------------------+---------+
|     aid|               title|                 url|      domain|votes|      user|          posted_at|comments|        source_title|         source_text|frontpage|
+--------+--------------------+--------------------+------------+-----+----------+-------------------+--------+--------------------+--------------------+---------+
|40177878|Show HN: Rule Spe...|https://gashlin.n...| gashlin.net|    1|       hcs|2024-04-27 06:52:49|       0|                NULL|🐑| 🐑  \n---|---...|    false|
|40168625|Bitcoin Will Powe...|https://www.coind...|coindesk.com|    1| PaulHoule|2024-04-26 12:33:29|       0|Bitcoin Will Powe...|Bitcoin Will Powe...|    false|
|40164119|Sen. Lummis: It'l...|https://www.coind...|coindesk.com|    2| PaulHoule|2024-04-25 23:01:23|       0|Sen. Lummis: It'l...|U.S. Sen. Lummis:...|    false|
|40205353|In the AI Economy...|https://reason.co...|  reason.com|    5|   jazzdev|2024-04-29 23:20:52|       3|In the AI Economy...|Will Artificial I...|     true|
|40205033|Wasabi Wallet-Dev...|https://www.coind...|coindesk.com|    1|paulpauper|2024-04-29 22:46:57|       0|Wasabi Wallet-Dev...|zkSNACKS, Develop...|    false|
+--------+--------------------+--------------------+------------+-----+----------+-------------------+--------+--------------------+--------------------+---------+
only showing top 5 rows
```

Figure 32: Data Collection

The column ”aid” is used to check the duplication and other possible check. Other features are planned to be used as the features. The ”frontpage” column will be marked as label in the final predictive model, with 3947 false and 751 true.



```
df.count()
```

```
4887
```

```
df2=df.dropDuplicates(["aid"])
df2.count()
```

```
4698
```

```
df2.write.parquet("hacker_news")
```

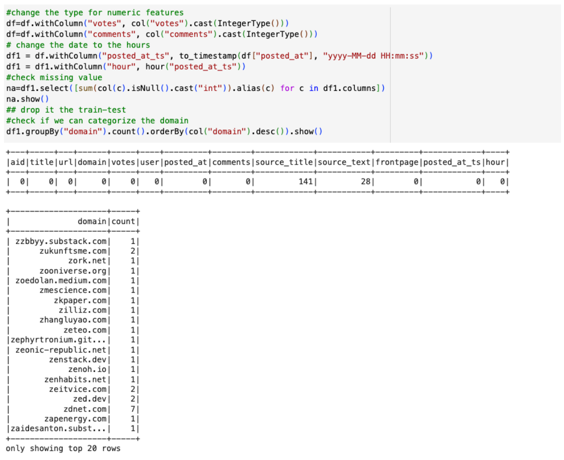
```
df2.groupBy("frontpage").count().orderBy(col("count").desc()).show()
```

```
+---------+-----+
|frontpage|count|
+---------+-----+
|    false| 3947|
|     true|  751|
+---------+-----+
```

Figure 33: Duplication Check

3.2 Data Preprocessing

After checking the types of columns, the ”comments” and ”votes” are transformed to numeric columns. The ”post_time” could also influence the time when the readers browse it. We assume that in the midnight, the readers may be less active than in the morning. Therefore, the hour is extracted to evaluate the moment for the readers. Missing value is based on the ”source_text” column where we will implement the text process. The ”domain” column as planned will be used to be a categorical feature to predict the model, but the result shows various results, finally we remove it for our final analysis. Considering the outline and the structure of the website of ”hacker news”, we extract 5 features to implement our models: ”title”, ”votes”,”comments”,”source_text”,”hour”(extracted from the ”post_at” time).

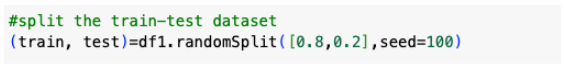


```
#change the type for numeric features
df=df.withColumn("votes", col("votes").cast(IntegerType()))
df=df.withColumn("comments", col("comments").cast(IntegerType()))
# change the date to the hours
df1 = df.withColumn("posted_at_ts", to_timestamp(df["posted_at"], "yyyy-MM-dd HH:mm:ss"))
df1 = df1.withColumn("hour", hour("posted_at_ts"))
#check missing value
na=df1.select([sum(col(c).isNull().cast("int")).alias(c) for c in df1.columns])
na.show()
## drop it the train-test
#check if we can categorize the domain
df1.groupBy("domain").count().orderBy(col("domain").desc()).show()
```

```
+---+-----+---+------+-----+----+---------+--------+------------+-----------+---------+------------+----+
|aid|title|url|domain|votes|user|posted_at|comments|source_title|source_text|frontpage|posted_at_ts|hour|
+---+-----+---+------+-----+----+---------+--------+------------+-----------+---------+------------+----+
|  0|    0|  0|     0|    0|   0|        0|       0|         141|         28|        0|           0|   0|
+---+-----+---+------+-----+----+---------+--------+------------+-----------+---------+------------+----+

+--------------------+-----+
|              domain|count|
+--------------------+-----+
| zzbbyy.substack.com|    1|
|      zukunftsme.com|    2|
|            zork.net|    1|
|      zooniverse.org|    1|
| zoedolan.medium.com|    1|
|      zmescience.com|    1|
|         zkpaper.com|    1|
|          zilliz.com|    1|
|      zhangluyao.com|    1|
|           zeteo.com|    1|
|zephyrtronium.git...|    1|
| zeonic-republic.net|    1|
|        zenstack.dev|    1|
|            zenoh.io|    1|
|         zenhabits.net|    1|
|        zeitvice.com|    2|
|             zed.dev|    2|
|           zdnet.com|    7|
|        zapenergy.com|    1|
|zaidesanton.subst...|    1|
+--------------------+-----+
only showing top 20 rows
```

Figure 34: Data Preprocess

In order to process the text, we at first to remove the news where there is no source text. Next, we label the frontpage as ”1” on the frontpage and ”0” not on the frontpage. In order to train the model, the data is split to train and test datas using the 80-20 approach in a random way.

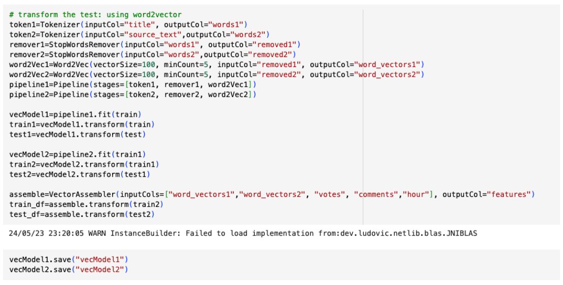


```
#split the train-test dataset
(train, test)=df1.randomSplit([0.8,0.2],seed=100)
```

Figure 35: Train-Test split

3.3 Text Processing

The processing of the text is applied to the columns of ”title” and ”source_text”. We use the Spark Mllib mostly to deal with the text. First we tokenize the title and the source text to get the words, then we remove the stop words to clear out the non-informative words, avoiding the redundancy. Then tf-idf and the word2vector two ways are taken into consideration to transform the text data. We capture that on the website, news title and content have some analogues and the sentiments,like ”Sorry Ipads”. Considering that the tf-idf is based on the frequency to calculate between-document similarity and classification, while the word2vector is targeted to the words analogue in a semantic level, which corresponds to the context characteristic on the website, the word2vector finally is applied to process the context. To transform the text data, we first planned to create one pipeline to all the text columns, but it seems that the spark ml don’t support it, so we build two pipelines to transform the title and context. Finally, the two vector models are saved to use predict the streaming news.



```
# transform the test: using word2vector
token1=Tokenizer(inputCol="title", outputCol="words1")
token2=Tokenizer(inputCol="source_text",outputCol="words2")
remover1=StopWordsRemover(inputCol="words1", outputCol="removed1")
remover2=StopWordsRemover(inputCol="words2",outputCol="removed2")
word2Vec1=Word2Vec(vectorSize=100, minCount=5, inputCol="removed1", outputCol="word_vectors1")
word2Vec2=Word2Vec(vectorSize=100, minCount=5, inputCol="removed2", outputCol="word_vectors2")
pipeline1=Pipeline(stages=[token1, remover1, word2Vec1])
pipeline2=Pipeline(stages=[token2, remover2, word2Vec2])

vecModel1=pipeline1.fit(train)
train1=vecModel1.transform(train)
test1=vecModel1.transform(test)

vecModel2=pipeline2.fit(train1)
train2=vecModel2.transform(train1)
test2=vecModel2.transform(test1)

assemble=VectorAssembler(inputCols=["word_vectors1","word_vectors2", "votes", "comments","hour"], outputCol="features")
train_df=assemble.transform(train2)
test_df=assemble.transform(test2)
```

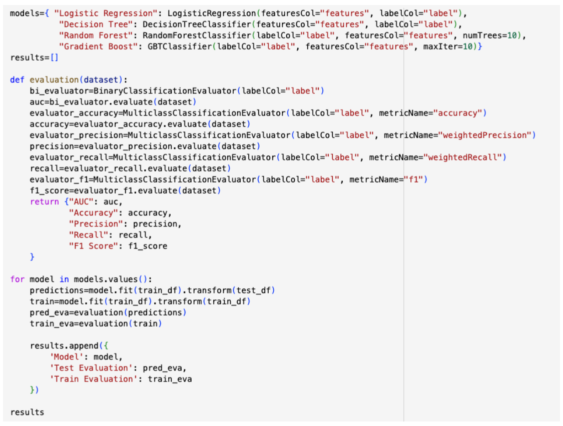
```
24/05/23 23:20:05 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.blas.JNIBLAS
```

```
vecModel1.save("vecModel1")
vecModel2.save("vecModel2")
```

Figure 36: Text Transform

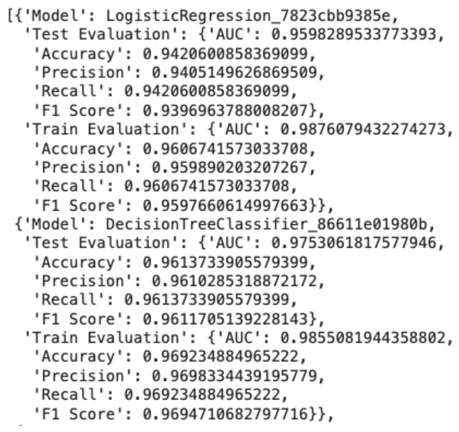
3.4 Model Fitting

After processing the text, we assemble the features: title_vector, context_vector, comments, votes, and hour to fit the model. Four models are fitted to train the model: logistic regression, decision tree, random forest and gradient boost. All the results shows good performance. The random forest and gradient boost shows the sign of overfitting of the training dataset. The decision tree and the logistic regression both perform very well and similarly. However, the logistic regression shows a more robust performance considering the differences between train and test metrics and also among the metrics themselves. At the same time, logistics regression makes the interpretation easier. It captured the patterns and make it more interpretive.



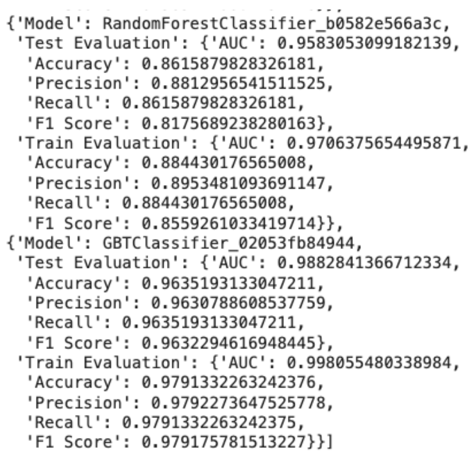
```
models={ "Logistic Regression": LogisticRegression(featuresCol="features", labelCol="label"),
          "Decision Tree": DecisionTreeClassifier(featuresCol="features", labelCol="label"),
          "Random Forest": RandomForestClassifier(labelCol="label", featuresCol="features", numTrees=10),
          "Gradient Boost": GBTClassifier(labelCol="label", featuresCol="features", maxIter=10)}
results=[]

def evaluation(dataset):
    bi_evaluator=BinaryClassificationEvaluator(labelCol="label")
    auc=bi_evaluator.evaluate(dataset)
    evaluator_accuracy=MulticlassClassificationEvaluator(labelCol="label", metricName="accuracy")
    accuracy=evaluator_accuracy.evaluate(dataset)
    evaluator_precision=MulticlassClassificationEvaluator(labelCol="label", metricName="weightedPrecision")
    precision=evaluator_precision.evaluate(dataset)
    evaluator_recall=MulticlassClassificationEvaluator(labelCol="label", metricName="weightedRecall")
    recall=evaluator_recall.evaluate(dataset)
    evaluator_f1=MulticlassClassificationEvaluator(labelCol="label", metricName="f1")
    f1_score=evaluator_f1.evaluate(dataset)
    return {"AUC": auc,
            "Accuracy": accuracy,
            "Precision": precision,
            "Recall": recall,
            "F1 Score": f1_score
    }

for model in models.values():
    predictions=model.fit(train_df).transform(test_df)
    train=model.fit(train_df).transform(train_df)
    pred_eva=evaluation(predictions)
    train_eva=evaluation(train)

    results.append({
        'Model': model,
        'Test Evaluation': pred_eva,
        'Train Evaluation': train_eva
    })

results
```

Figure 37: Model Fitting



```
[{'Model': LogisticRegression_7823cbb9385e,
  'Test Evaluation': {'AUC': 0.9598289533773393,
   'Accuracy': 0.9420600858369099,
   'Precision': 0.9405149626869509,
   'Recall': 0.9420600858369099,
   'F1 Score': 0.9396963788008207},
  'Train Evaluation': {'AUC': 0.9876079432274273,
   'Accuracy': 0.9606741573033708,
   'Precision': 0.959890203207267,
   'Recall': 0.9606741573033708,
   'F1 Score': 0.9597660614997663}},
 {'Model': DecisionTreeClassifier_86611e01980b,
  'Test Evaluation': {'AUC': 0.9753061817577946,
   'Accuracy': 0.9613733905579399,
   'Precision': 0.9610285318872172,
   'Recall': 0.9613733905579399,
   'F1 Score': 0.9611705139228143},
  'Train Evaluation': {'AUC': 0.9855081944358802,
   'Accuracy': 0.969234884965222,
   'Precision': 0.9698334439195779,
   'Recall': 0.969234884965222,
   'F1 Score': 0.9694710682797716}},
```

(a) Model Evaluation 1

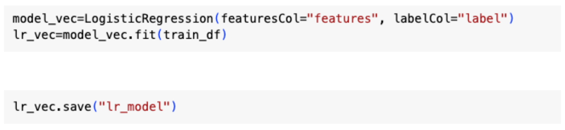


```
{'Model': RandomForestClassifier_b0582e566a3c,
 'Test Evaluation': {'AUC': 0.9583053099182139,
  'Accuracy': 0.8615879828326181,
  'Precision': 0.8812956541511525,
  'Recall': 0.8615879828326181,
  'F1 Score': 0.8175689238280163},
 'Train Evaluation': {'AUC': 0.9706375654495871,
  'Accuracy': 0.884430176565008,
  'Precision': 0.8953481093691147,
  'Recall': 0.884430176565008,
  'F1 Score': 0.8559261033419714}},
{'Model': GBTClassifier_02053fb84944,
 'Test Evaluation': {'AUC': 0.9882841366712334,
  'Accuracy': 0.9635193133047211,
  'Precision': 0.9630788608537759,
  'Recall': 0.9635193133047211,
  'F1 Score': 0.9632294616948445},
 'Train Evaluation': {'AUC': 0.998055480338984,
  'Accuracy': 0.9791332263242376,
  'Precision': 0.9792273647525778,
  'Recall': 0.9791332263242375,
  'F1 Score': 0.979175781513227}}]
```

(b) Model Evaluation 2

Figure 38: Model Evaluation

From the results, we can see that the gradient boost actually has the best performance, but it could be prone to overfitting for the train data. Random forest shows a bigger difference between train dataset and test dataset, it could be also caused by the overfitting. The decision tree in the test dataset perform slightly worse than the logistic regression. Considering the precision and recall, also the intepretibility, the logistic regression is chosen to be the final model, saved for the further streaming prediction.



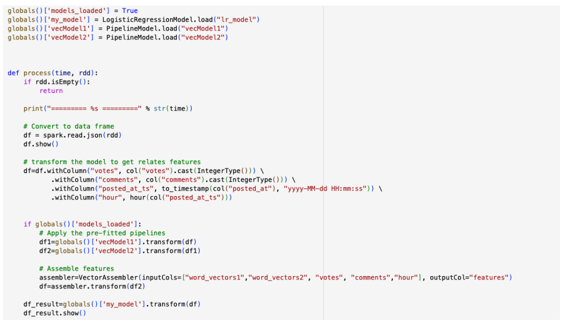
```
model_vec=LogisticRegression(featuresCol="features", labelCol="label")
lr_vec=model_vec.fit(train_df)
```

```
lr_vec.save("lr_model")
```

Figure 39: Model Save

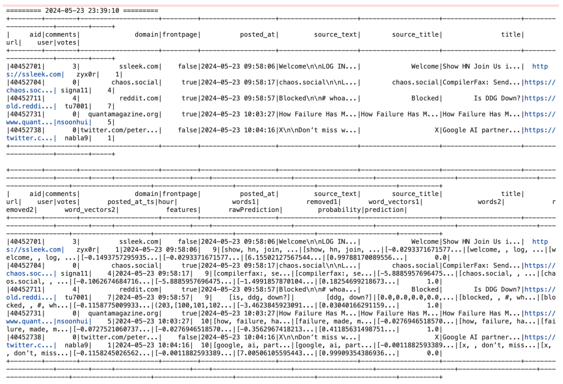
3.5 Streaming Prediction

Finally we apply our logistic regression model to the streaming news to predict if the news will be on the frontpage, which shows a good prediction of the frontpage. To apply our model into the streaming news, we need to transform the feature as we did to the numeric features, and also the texts. By using the saved model of the word-to-vector models and logistic regression model, the texts are transformed to the vectors, and we finally get the results of the prediction.



```python
globals()['models_loaded'] = True
globals()['my_model'] = LogisticRegressionModel.load("lr_model")
globals()['vecModel1'] = PipelineModel.load("vecModel1")
globals()['vecModel2'] = PipelineModel.load("vecModel2")



def process(time, rdd):
    if rdd.isEmpty():
        return

    print("========= %s =========" % str(time))

    # Convert to data frame
    df = spark.read.json(rdd)
    df.show()

    # transform the model to get relates features
    df=df.withColumn("votes", col("votes").cast(IntegerType())) \
           .withColumn("comments", col("comments").cast(IntegerType())) \
           .withColumn("posted_at_ts", to_timestamp(col("posted_at"), "yyyy-MM-dd HH:mm:ss")) \
           .withColumn("hour", hour(col("posted_at_ts")))


    if globals()['models_loaded']:
        # Apply the pre-fitted pipelines
        df1=globals()['vecModel1'].transform(df)
        df2=globals()['vecModel2'].transform(df1)

        # Assemble features
        assembler=VectorAssembler(inputCols=["word_vectors1","word_vectors2", "votes", "comments","hour"], outputCol="features")
        df=assembler.transform(df2)

    df_result=globals()['my_model'].transform(df)
    df_result.show()
```

Figure 40: Streaming Text Transform



```
========= 2024-05-23 23:39:10 =========
+--------+--------+--------------------+---------+-------------------+--------------------+--------------------+--------------------+--------
------------+--------+-----+
|     aid|comments|              domain|frontpage|          posted_at|         source_text|        source_title|               title|
url|    user|votes|
+--------+--------+--------------------+---------+-------------------+--------------------+--------------------+--------------------+--------
------------+--------+-----+
|40452701|       3|          ssleek.com|    false|2024-05-23 09:58:06|Welcome\n\nLOG IN...|             Welcome|Show HN Join Us i...|  http
s://ssleek.com|   zyx0r|    1|
|40452704|       0|        chaos.social|     true|2024-05-23 09:58:17|chaos.social\n\nL...|        chaos.social|CompilerFax: Send...|https://
chaos.soc...| signa11|    4|
|40452711|       4|          reddit.com|     true|2024-05-23 09:58:57|Blocked\n\n# whoa...|             Blocked|        Is DDG Down?|https://
old.reddi...|  tu7001|    7|
|40452731|       0|  quantamagazine.org|     true|2024-05-23 10:03:27|How Failure Has M...|How Failure Has M...|How Failure Has M...|https://
www.quant...|nsoonhui|    5|
|40452738|       0|twitter.com/peter...|    false|2024-05-23 10:04:16|X\n\nDon't miss w...|                   X|Google AI partner...|https://
twitter.c...|  nabla9|    1|
+--------+--------+--------------------+---------+-------------------+--------------------+--------------------+--------------------+--------
------------+--------+-----+

+--------+--------+--------------------+---------+-------------------+--------------------+--------------------+--------------------+--------
------------+--------+-----+-------------------+----+--------------------+--------------------+--------------------+--------------------+----
----------------+--------------------+--------------------+--------------------+--------------------+----------+
|     aid|comments|              domain|frontpage|          posted_at|         source_text|        source_title|               title|
url|    user|votes|       posted_at_ts|hour|              words1|            removed1|       word_vectors1|              words2|            r
emoved2|       word_vectors2|            features|       rawPrediction|         probability|prediction|
+--------+--------+--------------------+---------+-------------------+--------------------+--------------------+--------------------+--------
------------+--------+-----+-------------------+----+--------------------+--------------------+--------------------+--------------------+----
----------------+--------------------+--------------------+--------------------+--------------------+----------+
|40452701|       3|          ssleek.com|    false|2024-05-23 09:58:06|Welcome\n\nLOG IN...|             Welcome|Show HN Join Us i...|  http
s://ssleek.com|   zyx0r|    1|2024-05-23 09:58:06|   9|[show, hn, join, ...|[show, hn, join, ...|[-0.0293371671577...|[welcome, , log, ...|[w
elcome, , log, ...|[-0.1493757295935...|[-0.0293371671577...|[6.15502127567544...|[0.99788170089556...|       0.0|
|40452704|       0|        chaos.social|     true|2024-05-23 09:58:17|chaos.social\n\nL...|        chaos.social|CompilerFax: Send...|https://
chaos.soc...| signa11|    4|2024-05-23 09:58:17|   9|[compilerfax:, se...|[compilerfax:, se...|[-5.8885957696475...|[chaos.social, , ...|[cha
os.social, , ...|[-0.1062674684716...|[-5.8885957696475...|[-1.4991857870104...|[0.18254699218673...|       1.0|
|40452711|       4|          reddit.com|     true|2024-05-23 09:58:57|Blocked\n\n# whoa...|             Blocked|        Is DDG Down?|https://
old.reddi...|  tu7001|    7|2024-05-23 09:58:57|   9|    [is, ddg, down?]|        [ddg, down?]|[0.0,0.0,0.0,0.0,...|[blocked, , #, wh...|[blo
cked, , #, wh...|[-0.1158775009933...|(203,[100,101,102...|[-3.4623845923091...|[0.03040166291159...|       1.0|
|40452731|       0|  quantamagazine.org|     true|2024-05-23 10:03:27|How Failure Has M...|How Failure Has M...|How Failure Has M...|https://
www.quant...|nsoonhui|    5|2024-05-23 10:03:27|  10|[how, failure, ha...|[failure, made, m...|[-0.0276946518570...|[how, failure, ha...|[fai
lure, made, m...|[-0.0727521060737...|[-0.0276946518570...|[-0.3562967418213...|[0.41185631498751...|       1.0|
|40452738|       0|twitter.com/peter...|    false|2024-05-23 10:04:16|X\n\nDon't miss w...|                   X|Google AI partner...|https://
twitter.c...|  nabla9|    1|2024-05-23 10:04:16|  10|[google, ai, part...|[google, ai, part...|[-0.0011882593389...|[x, , don't, miss...|[x,
, don't, miss...|[-0.1158245026562...|[-0.0011882593389...|[7.00506105595443...|[0.99909354386936...|       0.0|
+--------+--------+--------------------+---------+-------------------+--------------------+--------------------+--------------------+--------
------------+--------+-----+-------------------+----+--------------------+--------------------+--------------------+--------------------+----
----------------+--------------------+--------------------+--------------------+--------------------+----------+
```

Figure 41: Streaming Prediction

3.6 Conclusion

The result in fact shows a good performance on the frontpage, by using the source content and title as text features, votes, comments and published hour as numeric features. The text are vectorized to fit the model. But there are some improvements for the further study: first, the time moment is important to readers, because readers are more active during the morning, lunch and before-sleeping time as users behavior in daily life, we could weight the time to the features to get a percentile ordinal feature to evaluate the result. Second, we planned to use the domain as the reference, but there are many categories, it could be quantified as the frequency in the data collection to evaluate the importance, also we could rank the domain, but there should be more documents and sources to support it. Third, for the text process, we use word-to-vector to get the result of of semantic analysis, in fact, the tf-idf could also be utilized for the topic modeling. At first we calculate the tf-idf, but it shows a huge bulks of the features, which is time-consuming. But the topic and the semantic analysis could be adopted at the same time, if the computer could handle, which may extract more crucial information from the text for the prediction.