Dive Into Anything

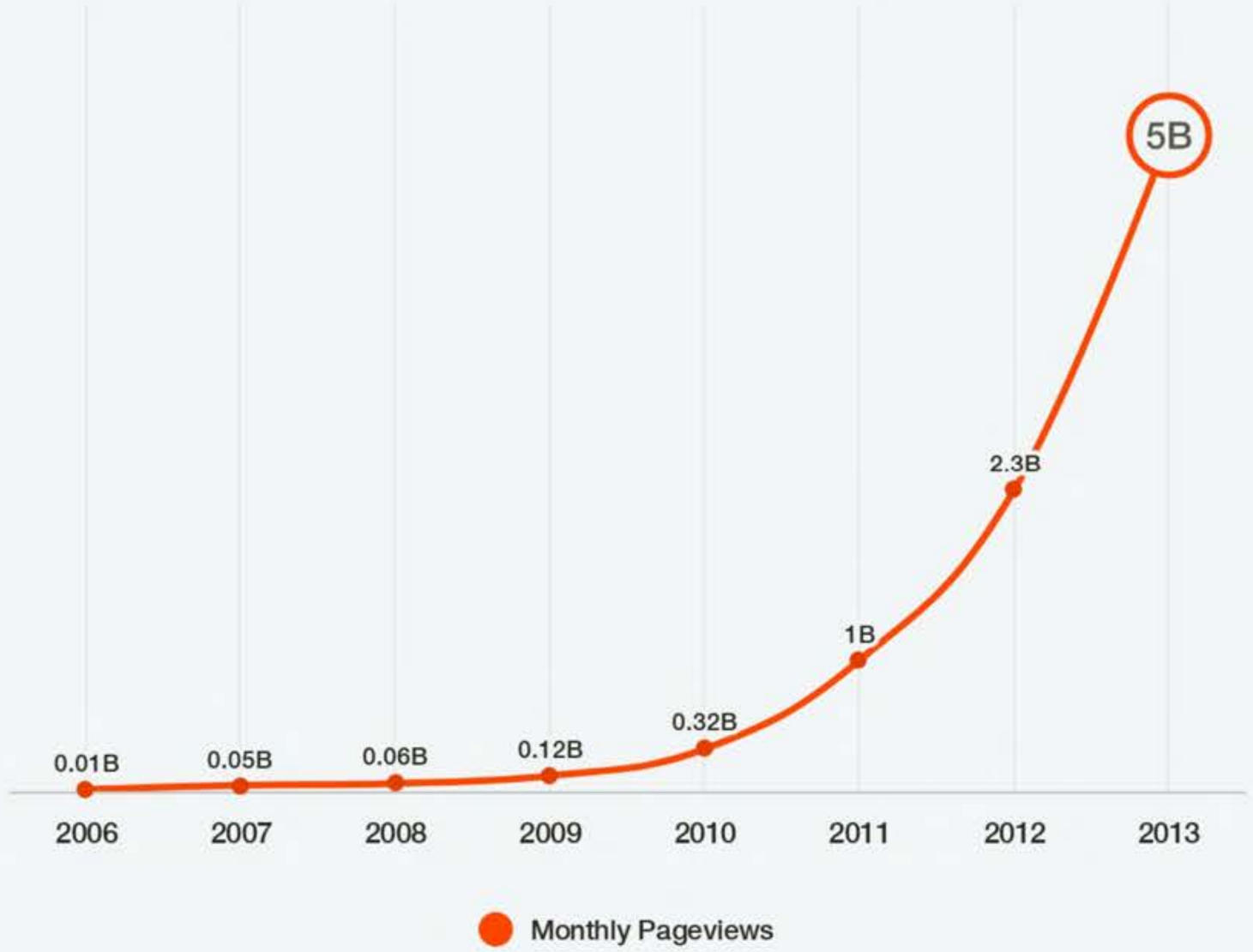
Classification of Subreddit Posts

Group 8

Aafrin Shehnaz, Shreenithi Sivakumar,
Shivram Sriramulu, Yogavarshni Ramachandran

Background

- Reddit is a platform where millions of users generate vast amounts of data daily across thousands of subreddits.
- Each subreddit is a niche community focused on a specific topic, from broad interests like news and technology to niche hobbies like underwater basket weaving.
- Reddit's data is a perfect example of 'big data,' characterized by its immense volume, high velocity, and wide variety.
- This data offers a unique opportunity to gain insights into digital human behavior, trends in content popularity, and the dynamics of online community engagement.

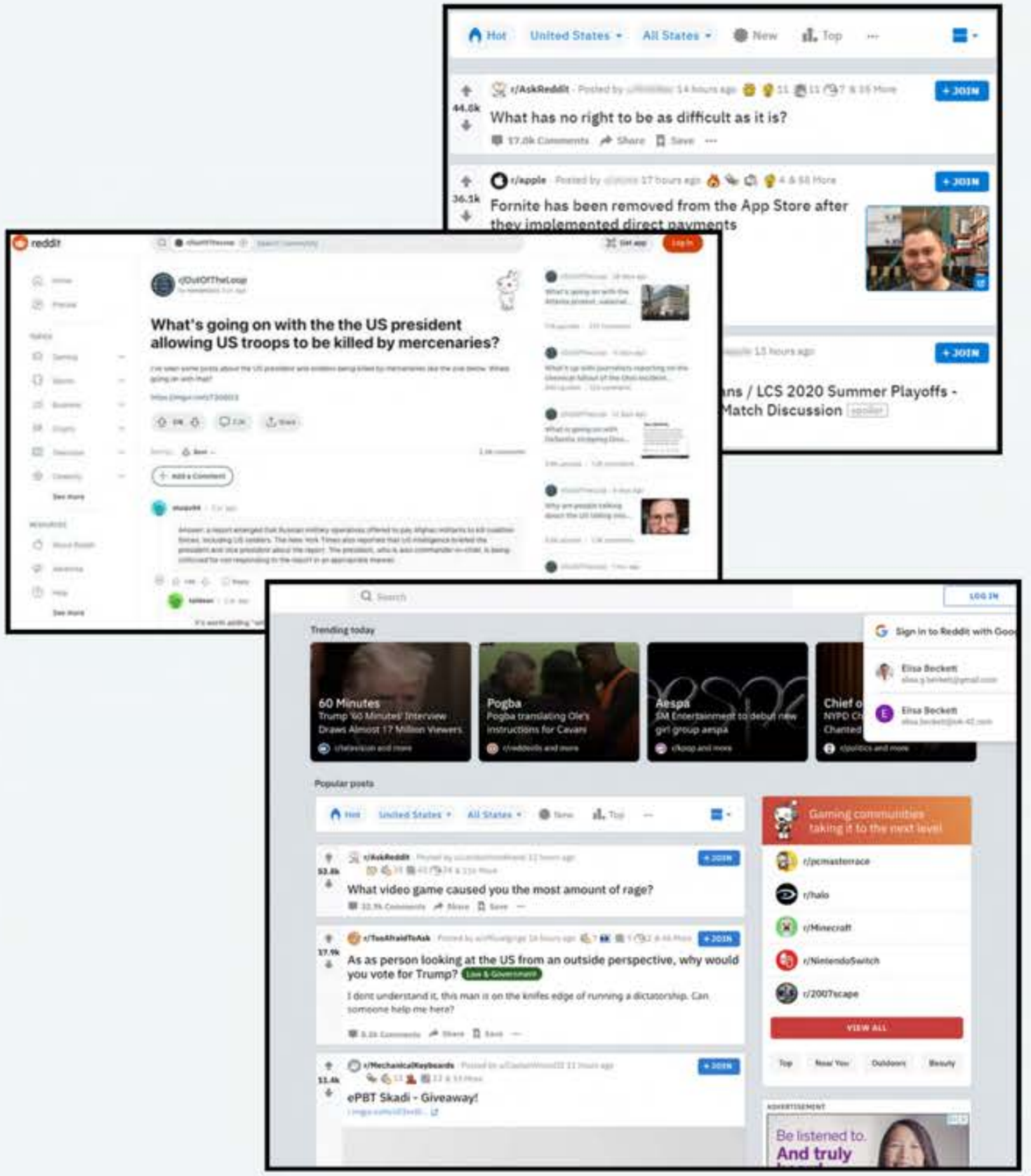


Reddit is the fastest growing social space in America.

It's also one of the most influential.

Problem Statement

Our project seeks to address the challenge of accurately classifying Reddit posts into their corresponding subreddits. As Reddit's content is both vast and diverse, effective categorization is crucial for optimizing user experience, facilitating content discovery, and fostering targeted engagement. Through the application of advanced machine learning techniques and big data analytics, we aim to develop an automated model capable of efficiently and accurately performing this classification, thereby enhancing the Reddit ecosystem



Subreddit Selection

r/cooking

3.9M subscribers
121 posts per day
2800 comments per day
3 million pageviews per month

r/programming

6M subscribers
200 posts per day
2760 comments per day
3.5 million pageviews per month

r/plants

346K subscribers
64 posts per day
550 comments per day
8200 pageviews per month

r/personalfinance

18.9 M subscribers
800 posts per day
8000 comments per day
12 million pageviews per month

r/travel

9.8M subscribers
450 posts per day
5620 comments per day
7.3 million pageviews per month

Big Data Technologies Used

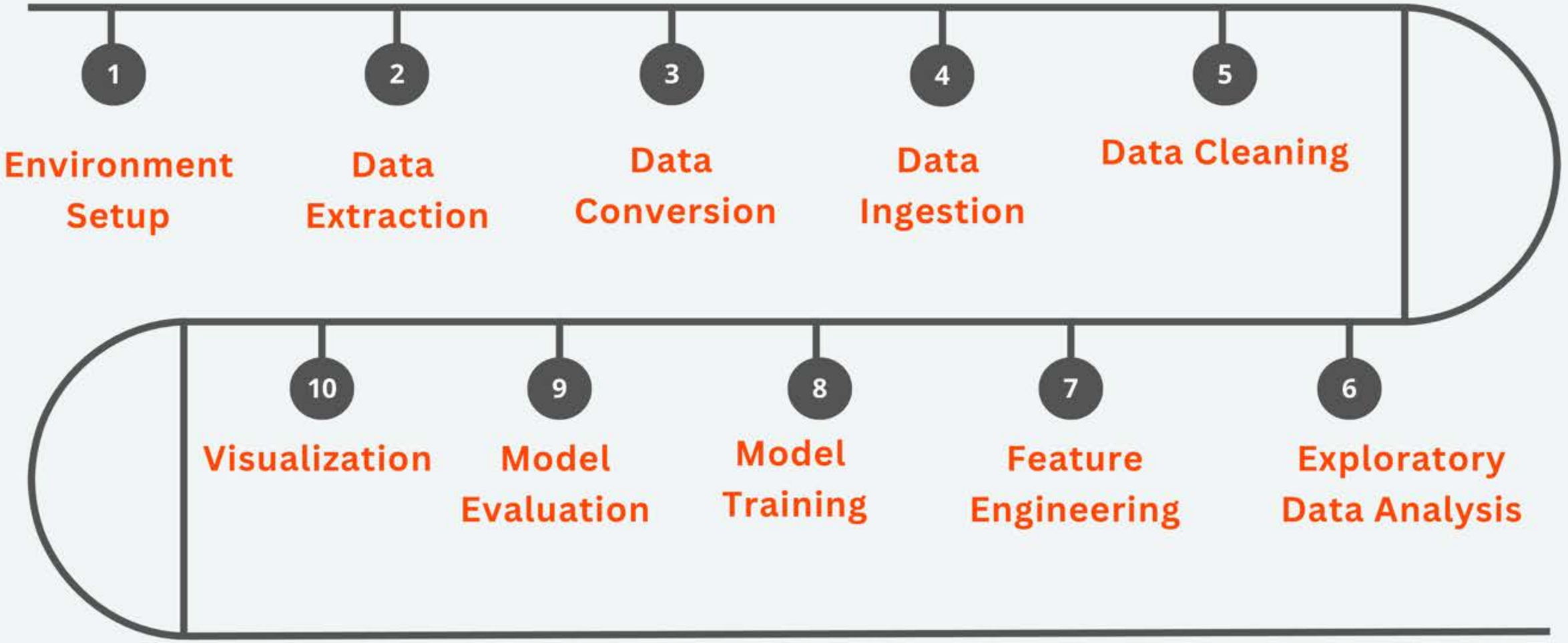
SPARK DATAFRAMES

Preprocessing and transformation of subreddit post data are facilitated through DataFrame transformations and actions, enhancing data quality before classification.

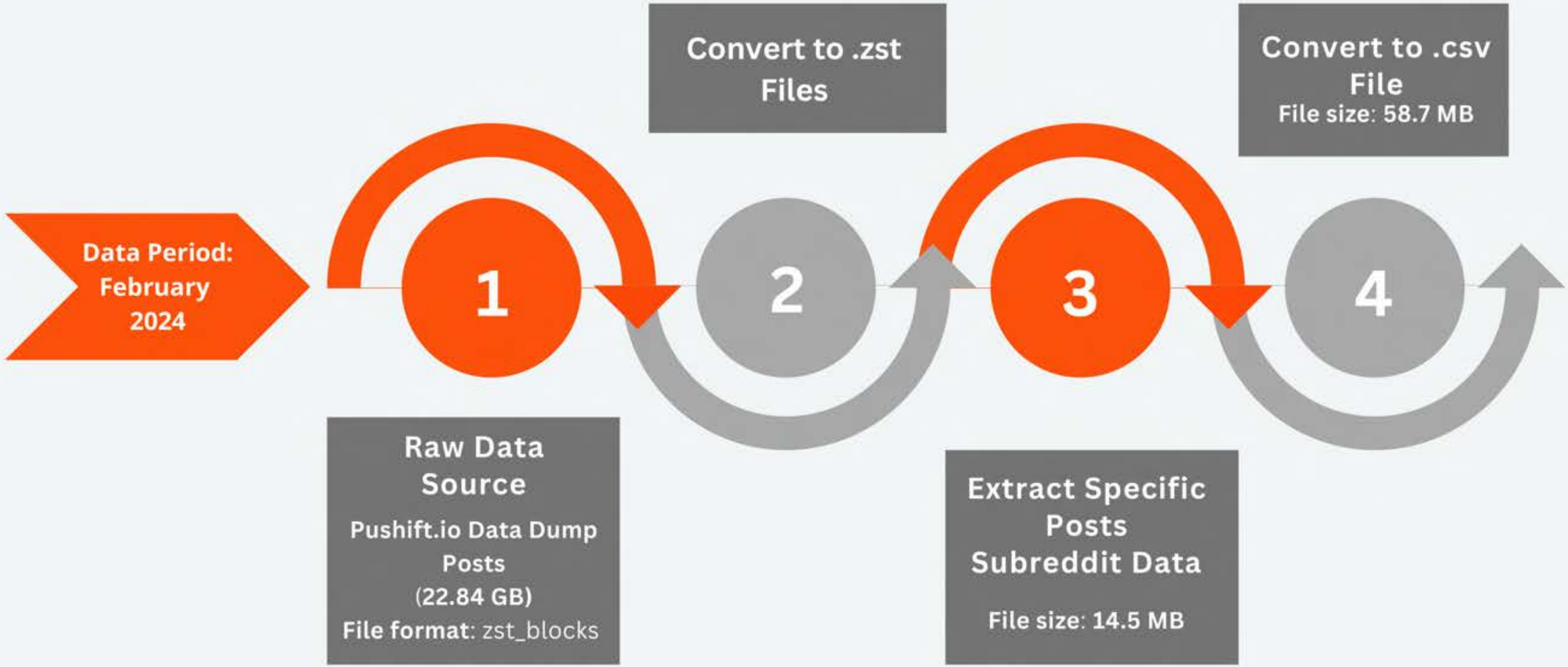
SPARK MLLIB

Empowers the building and training of classification models for predicting subreddit categories based on post content.

Methodology



Data Extraction & Conversion



Data Cleaning

Original dataframe shape

```
df1 = pd.read_csv(input_file, low_memory=False)
df1.shape
```

```
(35444, 118)
```

Manually Selected Columns: 16

```
df1.columns
```

```
Index(['_meta', 'all_awardings', 'allow_live_comments', 'approved_at_utc',
       'approved_by', 'archived', 'author', 'author_flair_background_color',
       'author_flair_css_class', 'author_flair_richtext',
       ...
       'is_gallery', 'link_flair_template_id', 'media_metadata',
       'url_overridden_by_dest', 'post_hint', 'preview', 'crosspost_parent',
       'crosspost_parent_list', 'author_cakeday', 'poll_data'],
      dtype='object', length=118)
```

```
selected_columns = [
    'author', 'created', 'domain', 'downs', 'link_flair_text',
    'num_comments', 'over_18', 'permalink', 'score', 'selftext',
    'subreddit_subscribers', 'title', 'ups', 'upvote_ratio', 'url','subreddit'
]
```

Dropped Redundant Columns: permalink, url, domain

Data Cleaning

```
processed_df.head()
```

	author	created	domain	link_flair_text	num_comments	over_18	score	selftext	subreddit	subreddit_subscribers	title	upvote_ratio
	Sorry_Lettuce_507	1706746097	self.travel	NaN	4	False	1	I'm from the U.S for whenever I travel it's ve...	travel	9372332	Can't decide where to go	0.60
				Images	0	False	1	NaN	travel	9372310	It a good day here	1.00
				Question	52	False	4	I'm going to hit Rome and sorrento for two wee...	travel	9372311	Italy advice... what to cut out? First trip	0.65
				Question	1	False	1	[removed]	travel	9372289	Should I take Dukoral before Mexico trip?	1.00
				Question	11	False	0	I do travel a lot but I am not a fan of touris...	travel	9372289	is there a map that I can find travel destinat...	0.44

```
# Transform over_18 to numbers
processed_df['over_18'] = processed_df['over_18'].astype(int)
```

```
processed_df['over_18']
```

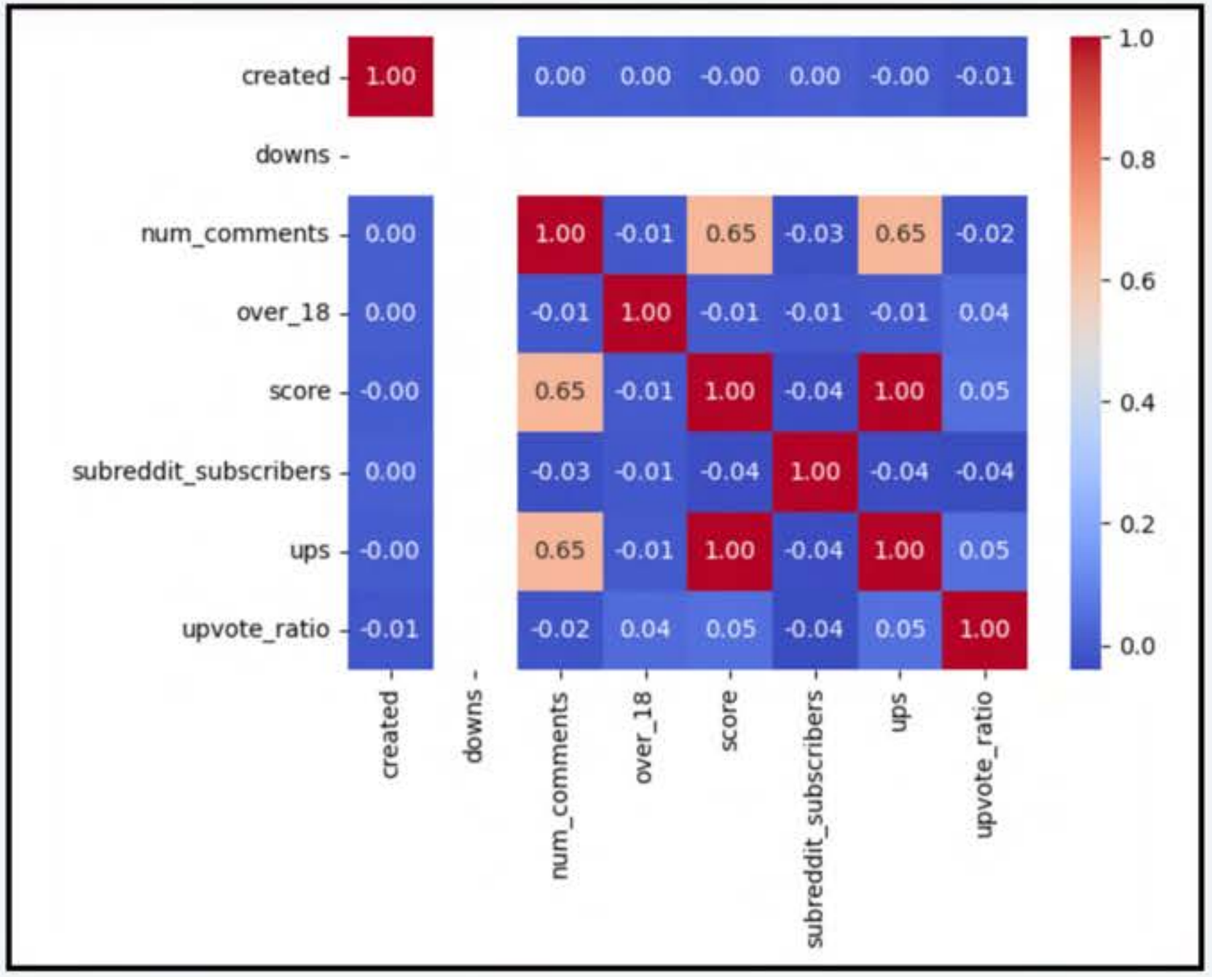
```
0        0
2        0
4        0
5        0
6        0
        ..
35439    0
35440    0
35441    0
35442    0
35443    0
Name: over_18, Length: 22404, dtype: int64
```

Row Cleaning: Convert "over_18" from Boolean to int

```
[removed]    7375
[deleted]      48
Name: selftext, dtype: int64
```

Row Cleaning: Remove [deleted], [removed] in "selftext "

Data Cleaning



Columns to remove:
"ups": Highly correlated with "score"
"downs": No values

Dropped Rows with subreddit = **programming**

```
processed_df['subreddit'].value_counts()

personalfinance    10231
travel              6106
Cooking             4124
plants              1943
programming          147
Name: subreddit, dtype: int64
```

Final Dataframe Size:

```
processed_df.shape

(22404, 9)
```

Data Cleaning

```
processed_df['link_flair_text'].value_counts()
```

```
Question                    3970
Other                       1418
Retirement                  1312
Debt                        1168
Taxes                       1096
Investing                    912
Open Discussion              873
Housing                      799
Auto                         752
Help                         712
Credit                       682
Recipe Request               558
Planning                     519
Budgeting                    493
Itinerary                    433
Saving                       420
Employment                   315
Insurance                    310
Food Safety                  211
Discussion                   210
Plant ID                     172
Images                        71
Recipe to Share               47
Success                       42
My Advice                     30
Third Party Horror Story      18
News                           4
Article                        3
Meta                           2
RESEARCH                       1
Trains                         1
Japan                          1
Advice Needed                  1
Recommendations                1
Transit Question               1
R1: Side income                1
R1: Help thread                1
R1: Poll or survey             1
Video                          1
Name: link_flair_text, dtype: int64
```

Null values in this will be replaced by “Unknown”

```
processed_df.isna().sum()
```

```
link_flair_text          4842
num_comments                0
over_18                     0
score                       0
selftext                    0
subreddit                   0
subreddit_subscribers       0
title                       0
upvote_ratio                0
dtype: int64
```

```
processed_df['link_flair_text'].fillna('Unknown', inplace=True)
```

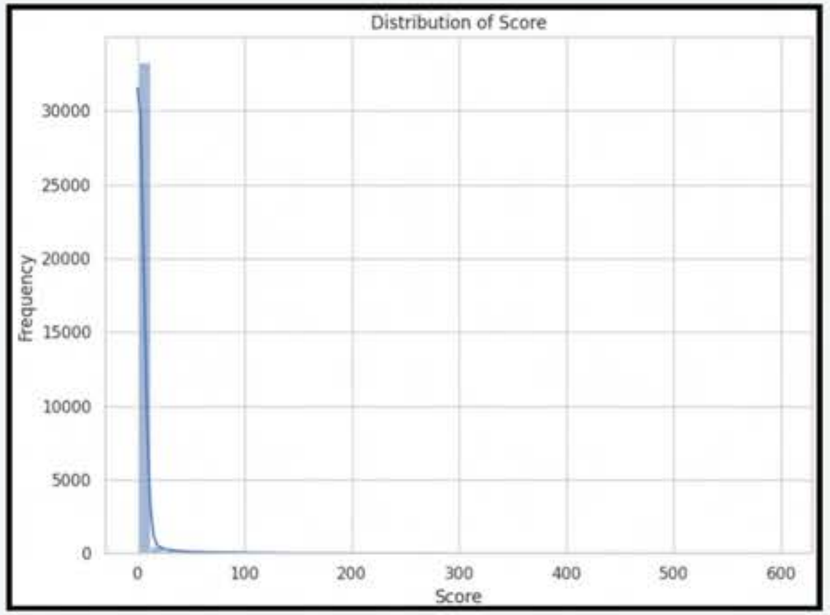
“link_flair_text” will be label encoded later

EDA - Summary Statistics

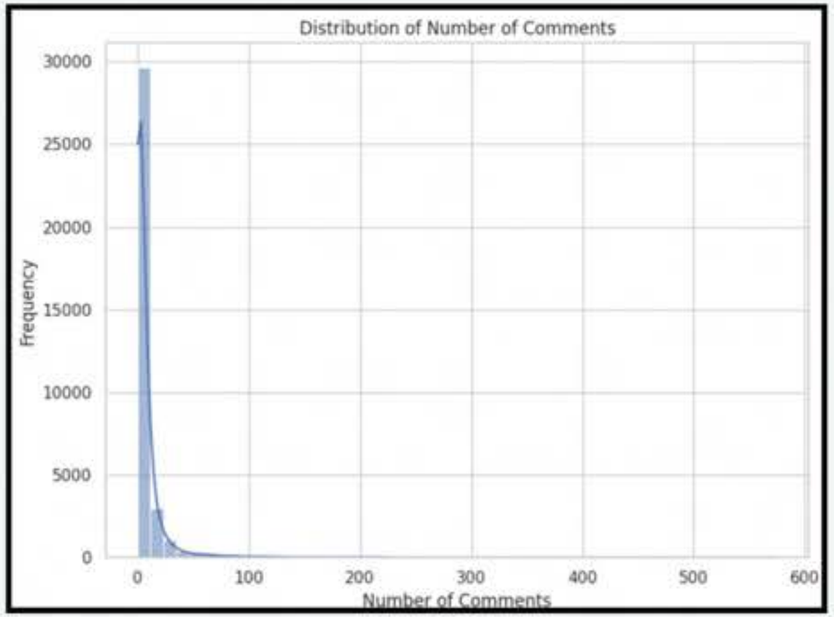
Dataset features have significantly different ranges of values. Normalize features into a uniform range

	num_comments	score	subreddit_subscribers	upvote_ratio
count	35444.000000	35444.000000	3.544400e+04	35444.000000
mean	12.203081	11.468344	1.110546e+07	0.767577
std	65.844124	109.156104	6.390045e+06	0.267794
min	0.000000	0.000000	3.380350e+05	0.030000
25%	0.000000	0.000000	5.909946e+06	0.500000
50%	2.000000	1.000000	9.516154e+06	0.920000
75%	7.000000	1.000000	1.870602e+07	1.000000
max	4333.000000	8336.000000	1.878354e+07	1.000000

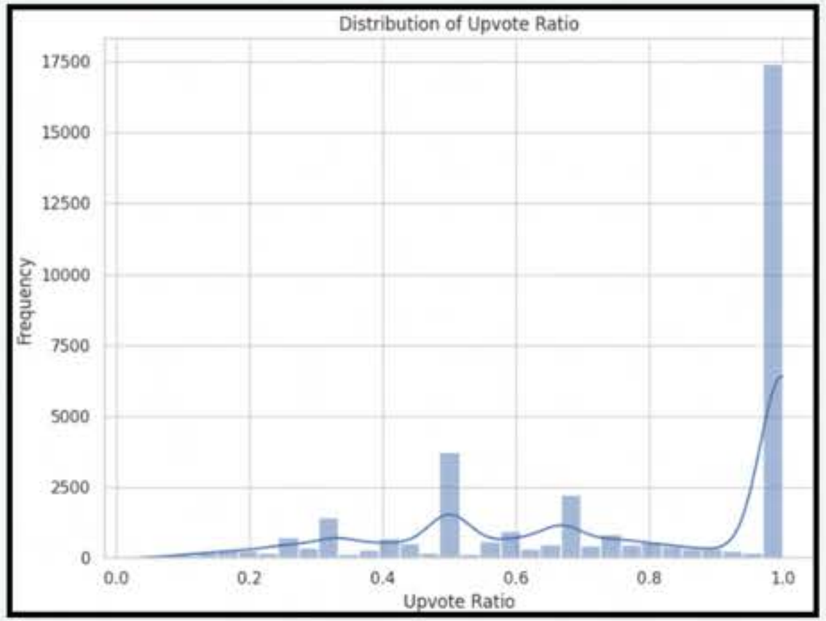
EDA - Numerical Analysis



Right skewed , most of the posts have a low number of comments.



Most of the posts have low scores.

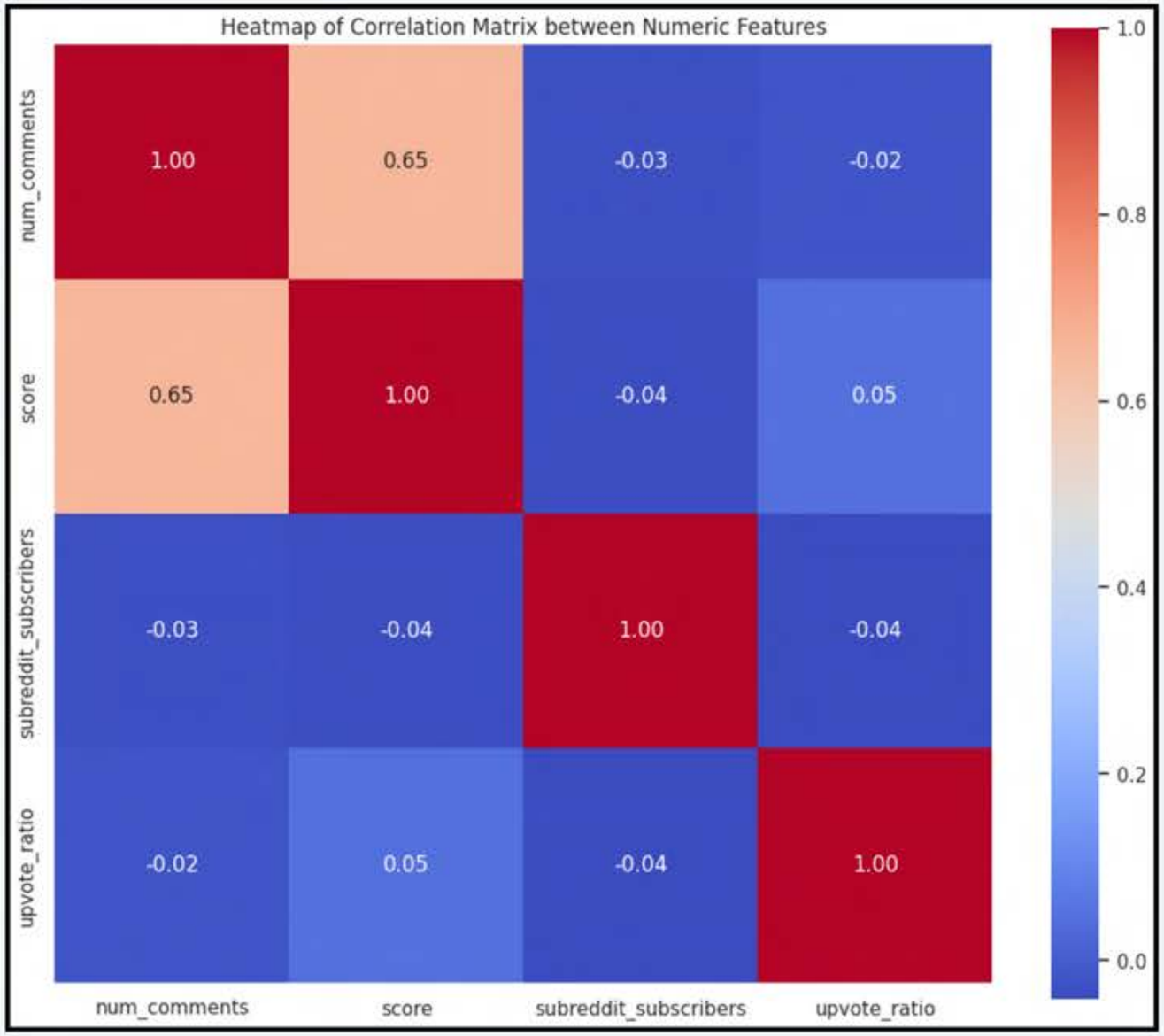


Most posts have an upvote ratio close to 1.

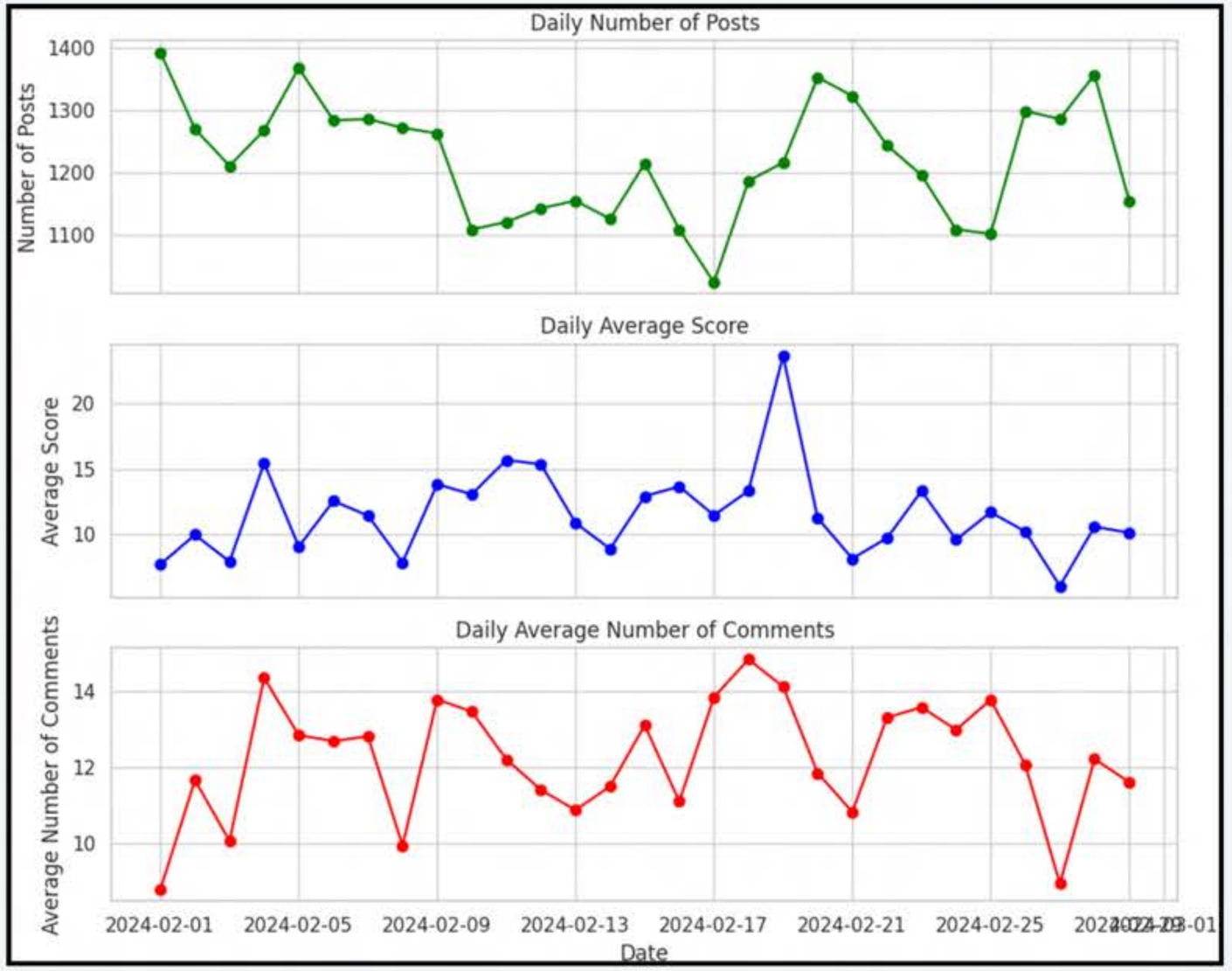
num_comments, score, upvote_ratio: outcomes of the subreddit, post hoc characteristics rather than predictive features, However, in some instances, they might capture the level of engagement or popularity that is typical for certain subreddits.

EDA - Correlation Analysis

- The positive correlation between num_comments and score which is 0.65, suggests that posts with more comments may also be the ones that receive good scores.
- The number of subscribers doesn't necessarily predict how many comments a post will get or its score.
- The upvote_ratio does not seem to have a strong linear relationship with the number of comments or the score of the posts.



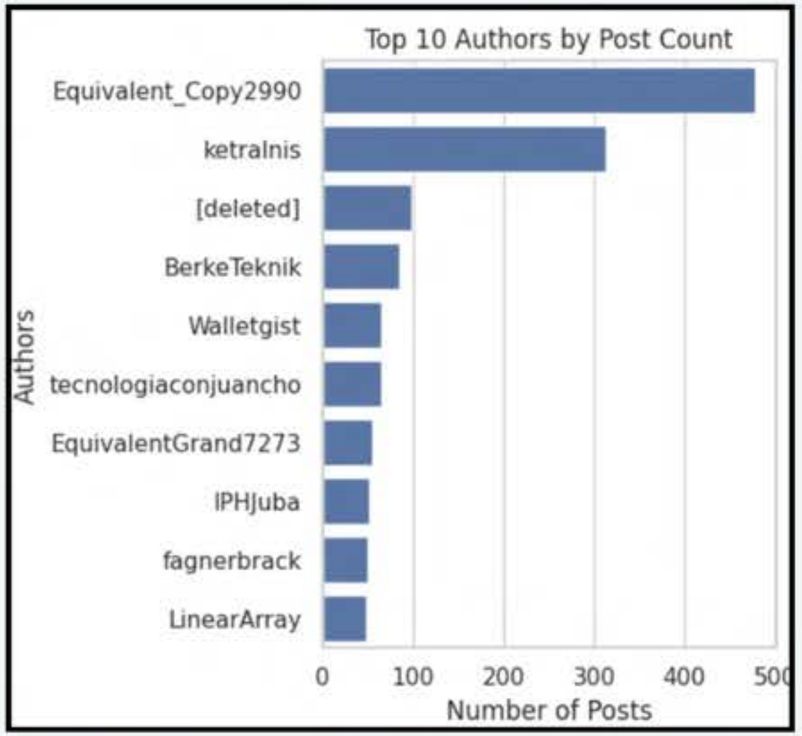
EDA - Date and Time Analysis



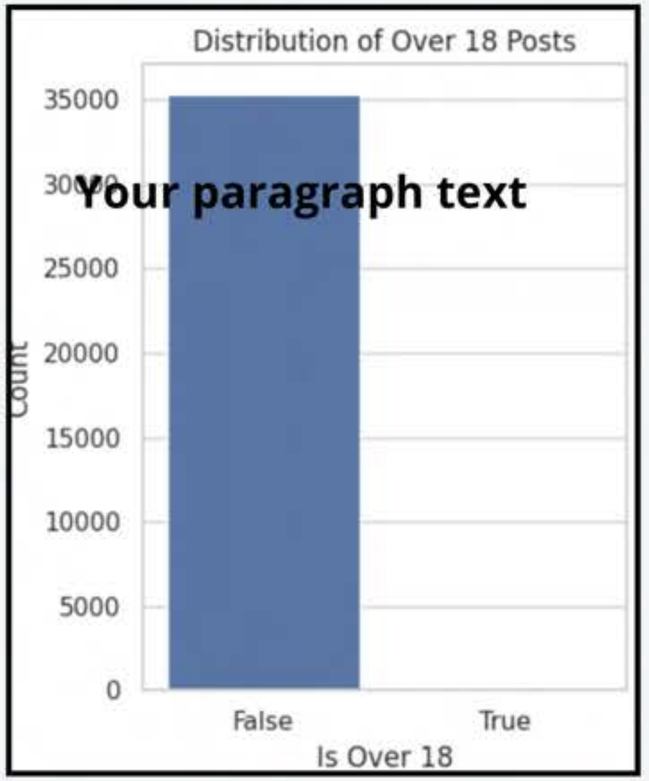
- **Daily Number of Posts**: There is no clear upward or downward trend, suggesting a relatively stable overall posting frequency
- **Daily Average Score:** The majority of days cluster around a certain average score range, with a few outliers.
- **Daily Average Number of Comments:** There's a fluctuating pattern in the average number of comments, indicating variability in user engagement over time.

created: Time-based features may provide some signal if posting patterns are different across subreddits, though this is likely to be less informative than the text itself for our classification task.

EDA - Categorical Analysis



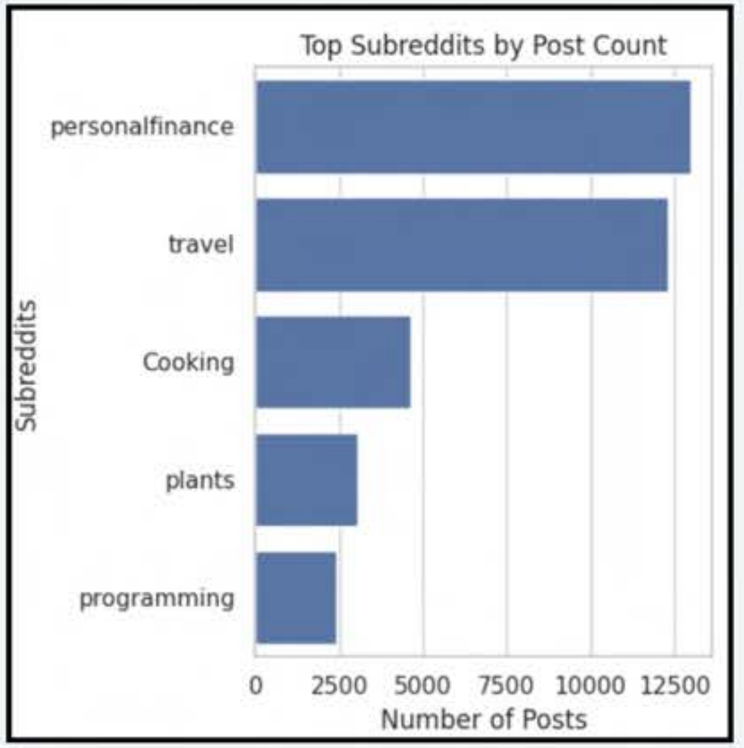

```
over_18
False    35327
True       117
Name: count, dtype: int64
```



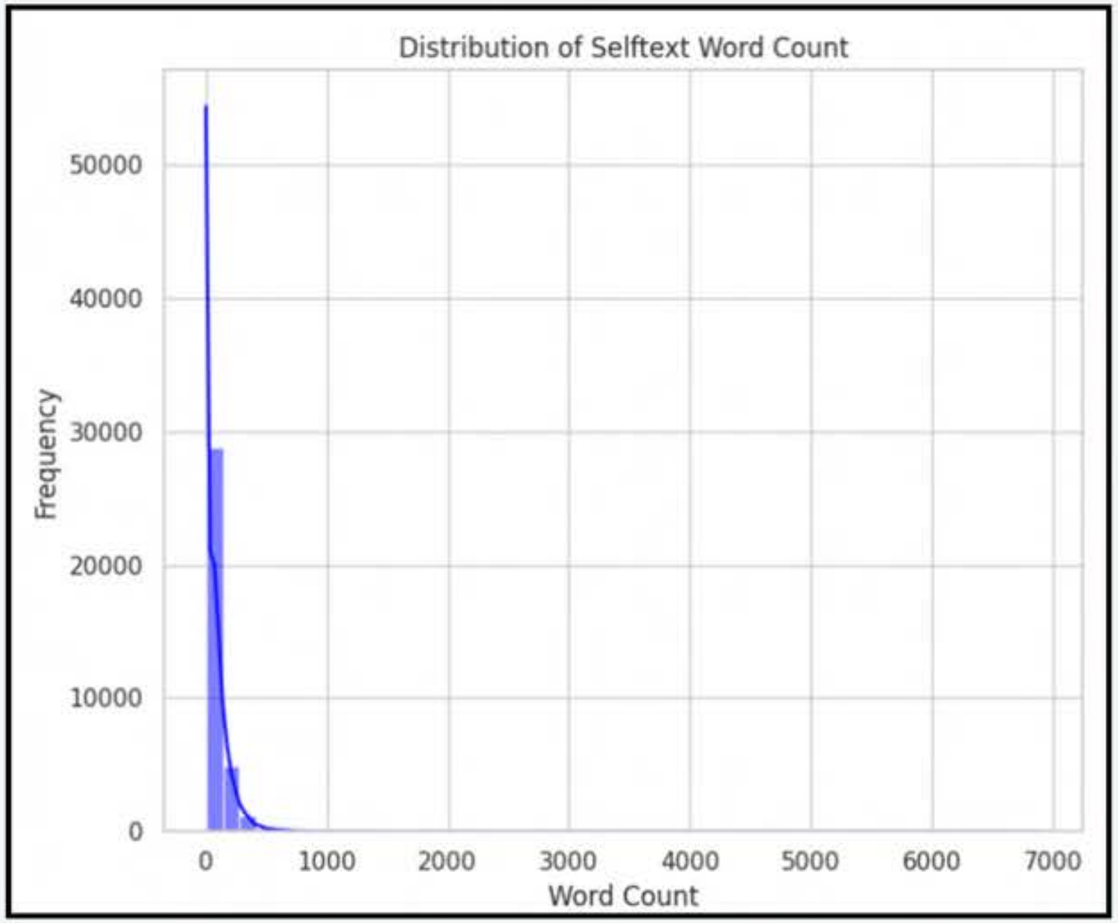
Your paragraph text

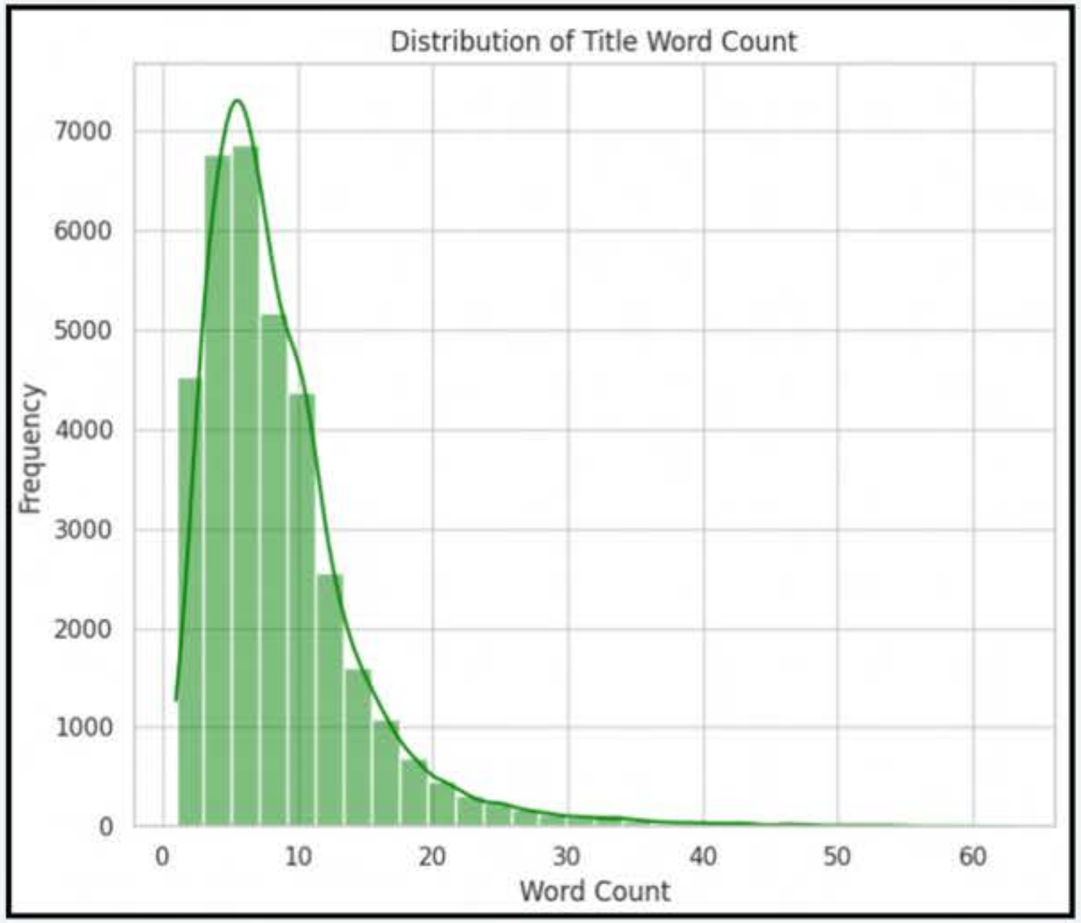
author: authors may post frequently to specific subreddits, it will not be a potential feature for our classification.

```
subreddit
personalfinance    12983
travel             12326
Cooking             4658
plants              3058
programming         2419
Name: count, dtype: int64
```



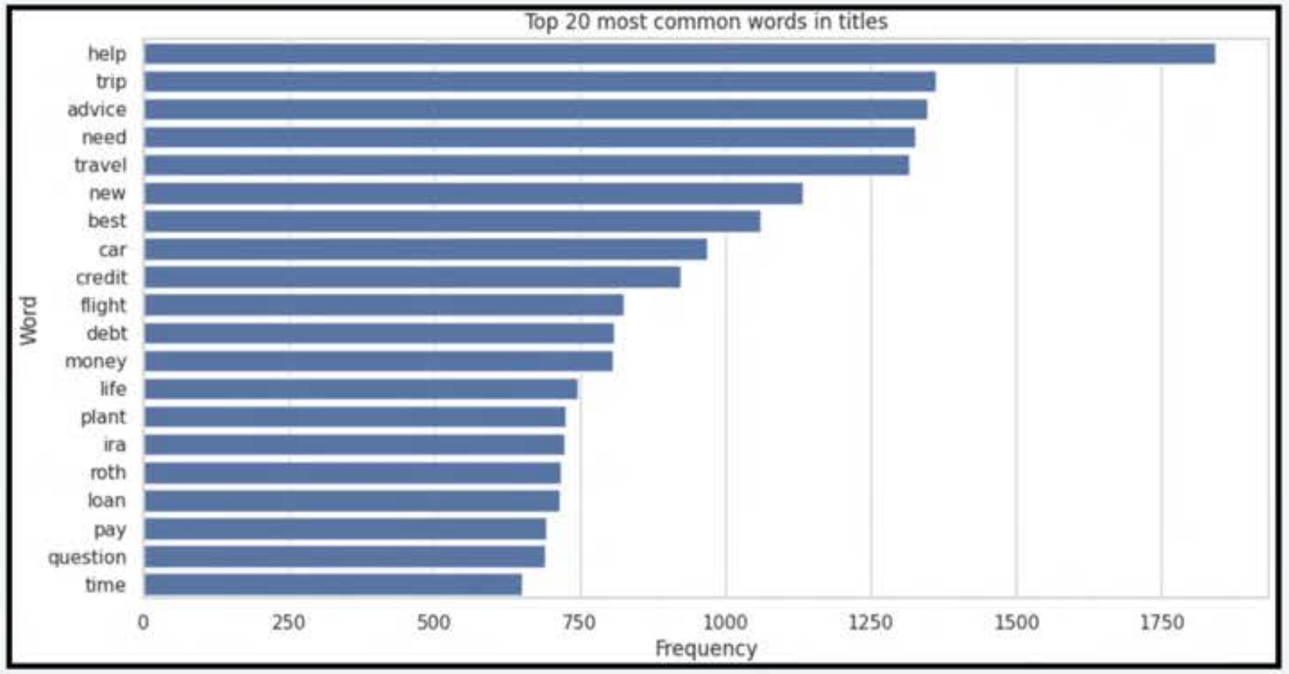
EDA - Text Data Analysis





- **Distribution of Selftext Word Count:** Very few entries have a high word count, suggesting that lengthy selftext posts are quite rare.
- **Distribution of Title Word Count:** The title word count appears normally distributed.Most titles are brief, typically around 5 to 11 words, aligning with the quartile values.

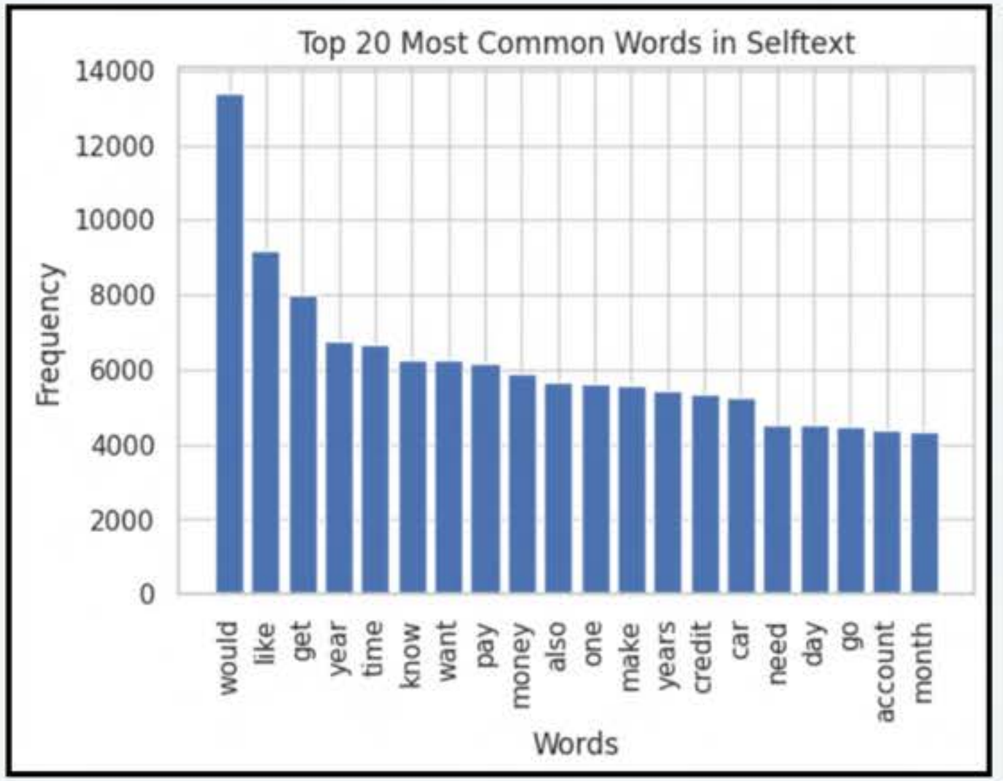
EDA - Text Data Analysis




```
common_trigrams

[(('credit', 'card', 'debt'), 435),
 (('would', 'greatly', 'appreciated'), 247),
 (('long', 'story', 'short'), 184),
 (('please', 'let', 'know'), 148),
 (('want', 'make', 'sure'), 133),
 (('high', 'yield', 'savings'), 125),
 (('would', 'make', 'sense'), 99),
 (('would', 'love', 'hear'), 94),
 (('student', 'loan', 'debt'), 93),
 (('max', 'roth', 'ira'), 91),
 (('pay', 'credit', 'card'), 89),
 (('yield', 'savings', 'account'), 83),
```

- **selftext**- the main body of the post is the most informative feature for this task
- **title** - summarizes the post and is rich in keywords that can be indicative of the subreddit.

```
common_bigrams

[(('roth', 'ira'), 1638),
 (('credit', 'card'), 1618),
 (('would', 'like'), 1163),
 (('last', 'year'), 879),
 (('first', 'time'), 817),
 (('feel', 'like'), 809),
 (('credit', 'score'), 796),
 (('thanks', 'advance'), 795),
 (('student', 'loans'), 759),
 (('years', 'ago'), 677),
```



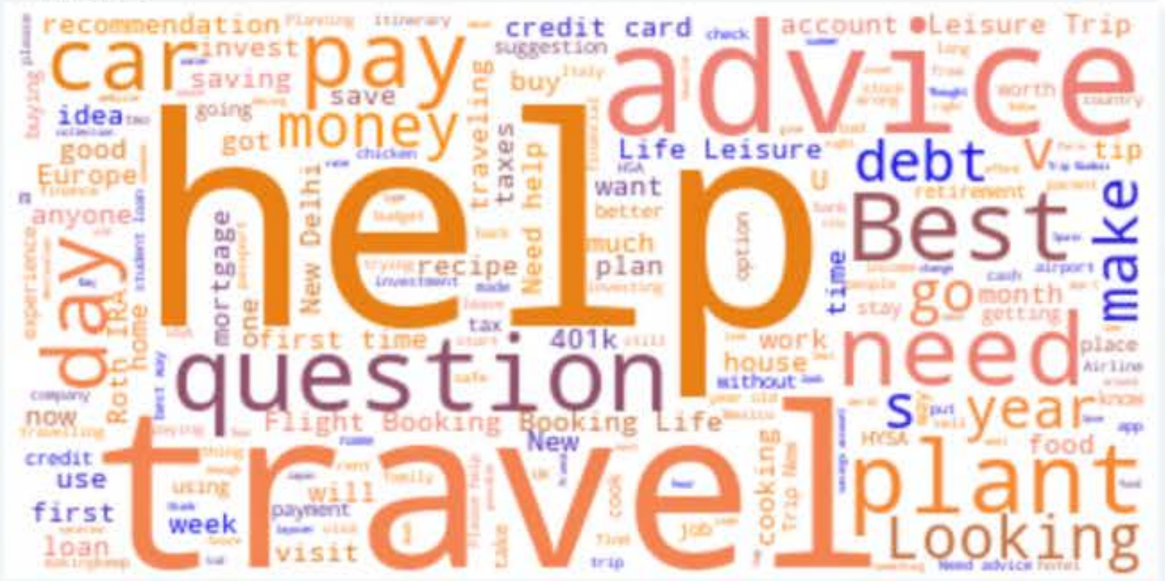
EDA - Text Data Analysis

Selftext

Selftext

Focus: "money," "house," "day," and "need"
Frequency: "money" and "need"
Diversity: "job," "credit card," "saving," "travel," and "retirement,"

Title



Title

Focus: "money," "plan," "need," "advice," and "year"
Frequency: "plan," "money," and "year"
Diversity: "Europe," "credit card," "retirement," and "food"

Data Splitting

- Stratified split to maintain class distribution similar to original dataset
- Datasets saved for direct loading into PySpark
- Data is split before transformation to avoid data leakage

Data Transformation and Pipeline Building (scikit-learn)

Text Processing

- Package Used: NLTK
- Remove stop words
- Lemmatization

Data Encoding

- Target: Label Encoder
- Categorical: Ordinal Encoder

Vectorization

- Text Columns: TF-IDF

Standardization

- Numeric Columns: StandardScaler

All other columns will be dropped.

Model Building and Hyperparameter Tuning

- **Models:** Random Forest, Logistic Regression, Decision Tree
- **Hyperparameter Tuning:** GridSearch CV in sklearn

```
# Define the pipeline
rf_pipeline = Pipeline([
    ('preprocessing', column_transformer),
    ('classifier', RandomForestClassifier(random_state=42))
])

# Grid Search
rf_param_grid = {
    'classifier__n_estimators': [100, 200, 300, 400],
    'classifier__max_depth': [10, 20, 30, None],
    'classifier__min_samples_split': [2, 5, 10]
}

rf_grid_search = GridSearchCV(rf_pipeline, rf_param_grid, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)
rf_grid_search.fit(X_train, y_train)
```

Sample code for Random Forest Model

Model Evaluation & Best Parameters

Random Forest

```
Fitting 5 folds for each of 48 candidates, totalling 240 fits
Best Parameters: {'classifier__max_depth': None, 'classifier__min_samples_split': 2, 'classifier__n_estimators': 400}
Best Score: 0.994707181565645
Validation Accuracy:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       618
           1       1.00      1.00      1.00      1535
           2       0.98      0.98      0.98       292
           3       0.99      0.99      0.99       916

    accuracy                           0.99      3361
   macro avg       0.99      0.99      0.99      3361
weighted avg       0.99      0.99      0.99      3361
```

Logistic Regression

```
Fitting 5 folds for each of 32 candidates, totalling 160 fits
/Users/shreenithi/anaconda3/lib/python3.11/site-packages/sklearn/svm/_base.py:1244: ConvergenceWarning: Liblinear failed to converge, increase the number of iterations.
  warnings.warn(
Best Parameters: {'classifier__C': 1, 'classifier__class_weight': 'balanced', 'classifier__max_iter': 200, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}
Best Score: 0.9977043255938014
Validation Accuracy:
              precision    recall  f1-score   support

           0       1.00      0.99      0.99       618
           1       1.00      1.00      1.00      1535
           2       0.99      0.99      0.99       292
           3       1.00      1.00      1.00       916

    accuracy                           1.00      3361
   macro avg       1.00      1.00      1.00      3361
weighted avg       1.00      1.00      1.00      3361
```

Decision Tree

```
Fitting 5 folds for each of 80 candidates, totalling 400 fits
Best Parameters: {'classifier__max_depth': None, 'classifier__min_samples_leaf': 1, 'classifier__min_samples_split': 2}
Best Score: 1.0
Validation Accuracy:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       618
           1       1.00      1.00      1.00      1535
           2       1.00      1.00      1.00       292
           3       1.00      1.00      1.00       916

    accuracy                           1.00      3361
   macro avg       1.00      1.00      1.00      3361
weighted avg       1.00      1.00      1.00      3361
```

Best Parameters will be used directly as Model Parameters in PySpark

PySpark - Data Loading

```
from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .config("spark.driver.host", "localhost") \
    .config("spark.driver.memory", "4g") \
    .config("spark.executor.memory", "4g") \
    .config("spark.executor.memoryOverhead", "1g") \
    .appName('Reddit Analysis') \
    .getOrCreate()

spark
```

```
24/04/30 13:07:26 WARN Utils: Your hostname, Shreenit
d (on interface en0)
24/04/30 13:07:26 WARN Utils: Set SPARK_LOCAL_IP if y
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
24/04/30 13:07:27 WARN NativeCodeLoader: Unable to lo
e
24/04/30 13:07:27 WARN Utils: Service 'SparkUI' could
```

SparkSession - in-memory

SparkContext

Spark UI

Version	v3.5.1
Master	local[*]
AppName	Reddit Analysis

Initiate spark session

Load pandas DataFrame into spark DataFrame

```
# Convert the pandas DataFrame to a PySpark DataFrame
train_df = spark.createDataFrame(p_train_df)
val_df = spark.createDataFrame(p_val_df)
test_df = spark.createDataFrame(p_test_df)


train_row_count = train_df.count()
# Count the number of columns
train_column_count = len(train_df.columns)

val_row_count = val_df.count()
# Count the number of columns
val_column_count = len(val_df.columns)

test_row_count = test_df.count()
# Count the number of columns
test_column_count = len(test_df.columns)



print(f"Shape of train_df : ({train_row_count}, {train_column_count})")
print(f"Shape of val_df : ({val_row_count}, {val_column_count})")
print(f"Shape of test_df : ({test_row_count}, {test_column_count})")
# print(f"Number of columns: {column_count}")
```

```
Shape of train_df : (15682, 9)
Shape of val_df : (3361, 9)
Shape of test_df : (3361, 9)
```

PySpark - Data Loading

```
# Show the PySpark DataFrames
train_df.show(5)
```

```
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
|link_flair_text|num_comments|over_18|score|            selftext|subreddit_subscribers|               title|upvote_ratio|      subreddit|
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
|       Question|           4|      0|    0|My birthday is in...|             9538962|Looking to travel...|        0.5|         travel|
|       Question|           6|      0|    0|I would appreciat...|             9454777|How can a person ...|        0.3|         travel|
|     Retirement|           2|      0|    1|**Question: Shoul...|            18675591|PreTax vs. Post-T...|        1.0|personalfinance|
|       Question|           4|      0|    1|will have total 1...|             9436096|Tips for London >...|        1.0|         travel|
|Open Discussion|           2|      0|    0|i need to deep fr...|             3902917|deep fry in coppe...|        0.5|        Cooking|
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
only showing top 5 rows
```

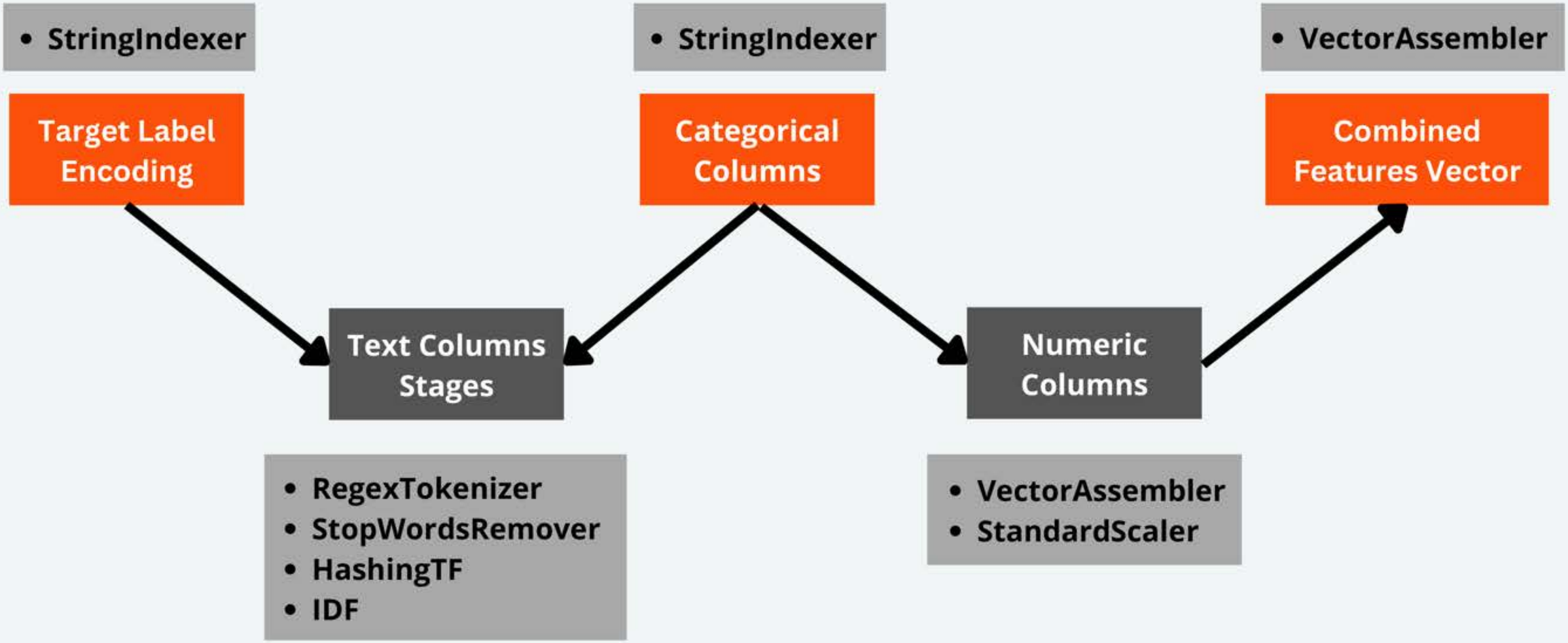
```
val_df.show(5)
```

```
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
|link_flair_text|num_comments|over_18|score|            selftext|subreddit_subscribers|               title|upvote_ratio|      subreddit|
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
|           Help|           2|      0|    3|Got these a while...|              339741|Plant Id and what...|        1.0|         plants|
|        Unknown|           8|      0|    0|Probably a pretty...|             3890359|Chicken stock advice|        0.5|        Cooking|
|     Employment|           2|      0|    1|Hi, Im 23 years o...|            18774319|personal finances...|       0.67|personalfinance|
|        Unknown|           2|      0|    0|I was gifted seve...|              340889|Would you tell so...|        0.5|         plants|
|       Question|           0|      0|    1|Hi everyone! \n\n...|             9535404|Seeking Advice fo...|        1.0|         travel|
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
only showing top 5 rows
```

```
test_df.show(5)
```

```
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
|link_flair_text|num_comments|over_18|score|            selftext|subreddit_subscribers|               title|upvote_ratio|      subreddit|
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
|     Retirement|          11|      0|    1|I have some money...|            18739859|Estimating MAGI f...|        1.0|personalfinance|
|        Unknown|          10|      0|    3|What are your fav...|             3895361|I have a new pie ...|       0.72|        Cooking|
|        Unknown|          10|      0|    0|I'm a pretty basi...|             3901417| Burgers… air fryer?|       0.33|        Cooking|
|        Unknown|           7|      0|    0|We bought ground ...|             3890701|How to tell bad meat|       0.33|        Cooking|
|         Images|           0|      0|    2|Hey everyone,\n\n...|             9600035|Qatar Airways Avi...|        1.0|         travel|
+---------------+------------+-------+-----+--------------------+--------------------+--------------------+-----------+---------------+
only showing top 5 rows
```

PySpark - Transformers and Pipeline Stages



PySpark - Model Building

- **Models:** Random Forest, Logistic Regression, Decision Tree
- **Hyperparameters:** Use best params from GridSearch CV in sklearn

```
# Classifier
rf_classifier = RandomForestClassifier(
    featuresCol="features",
    labelCol="label",
    numTrees=400,
    maxBins=40
)

# Build the pipeline
rf_all_stages = [stage for col in text_cols for stage in create_text_pipeline_stages(col)] + cat_indexers + [assembler, scaler, feature_assemb
rf_pipeline = Pipeline(stages=rf_all_stages)

# Fit the pipeline
rf_model = rf_pipeline.fit(train_df_indexed)

# Transform validation and test datasets
rf_val_predictions = rf_model.transform(val_df_indexed)
rf_test_predictions = rf_model.transform(test_df_indexed)
```

Sample code for Random Forest Model

PySpark - Model Evaluation

Pacakage Used:
MulticlassClassificationEvaluator

```
---------------------------
Random Forest Evaluation
---------------------------
Validation accuracy: 0.89
Test accuracy: 0.89
F1 Score: 0.8782359489944733
Precision: 0.8937932151985946
Recall: 0.8914013686402856
```

```
---------------------------
Decision Tree Evaluation
---------------------------
Validation accuracy: 0.99
Test accuracy: 0.99
F1 Score: 0.9931600365651072
Precision: 0.993337622522195
Recall: 0.9931567985718537
```

```
-------------------------------
Logistic Regression Evaluation
-------------------------------
Validation accuracy: 0.68
Test accuracy: 0.67
F1 Score: 0.630591715003937
Precision: 0.6197493067334993
Recall: 0.6843201428146386
```

Vectorization

```
from pyspark.ml.feature import HashingTF, IDF, Tokenizer
# Apply Tokenizer to tokenize the title column
tokenizer = Tokenizer(inputCol="title", outputCol="words")
words_data = tokenizer.transform(scaled_df_normalized)

# Apply HashingTF to convert words to feature vectors
hashing_tf = HashingTF(inputCol="words", outputCol="rawFeatures"
featurized_data = hashing_tf.transform(words_data)

# Apply IDF to scale the raw features
idf = IDF(inputCol="rawFeatures", outputCol="tf_idf_features")
idf_model = idf.fit(featurized_data)
tf_idf_data = idf_model.transform(featurized_data)

# Show the DataFrame with TF-IDF features
tf_idf_data.select("title", "tf_idf_features").show(5)
```

Applying One hot encoding to Target Feature

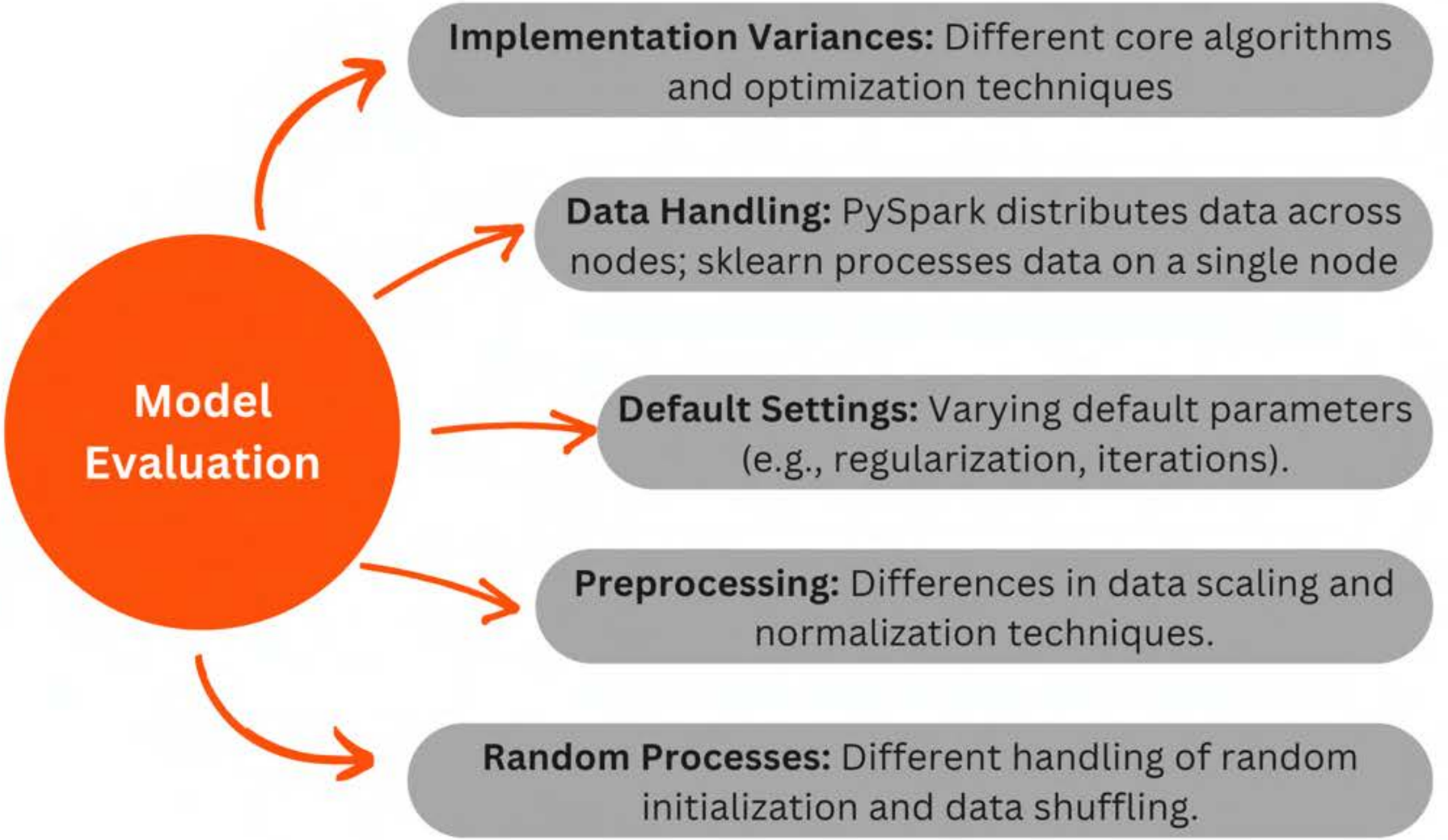
```
from pyspark.ml.feature import StringIndexer, OneHotEncoder
from pyspark.ml import Pipeline

# StringIndexer to convert subreddit column to numeric labels
indexer = StringIndexer(inputCol="subreddit", outputCol="label")
indexed_df = indexer.fit(tf_idf_data).transform(tf_idf_data)

# OneHotEncoder to convert numeric labels to one-hot encoded vectors
encoder = OneHotEncoder(inputCol="label", outputCol="encoded_label")
encoded_df = encoder.fit(indexed_df).transform(indexed_df)

# Show the DataFrame with one-hot encoded labels
encoded_df.select("subreddit", "label", "encoded_label").show(5)
```

```
+---------+-----+-------------+
|subreddit|label|encoded_label|
+---------+-----+-------------+
|   travel|  1.0|(4,[1],[1.0])|
|   travel|  1.0|(4,[1],[1.0])|
|   travel|  1.0|(4,[1],[1.0])|
|   travel|  1.0|(4,[1],[1.0])|
|   travel|  1.0|(4,[1],[1.0])|
+---------+-----+-------------+
```

ML MODELS - Text only

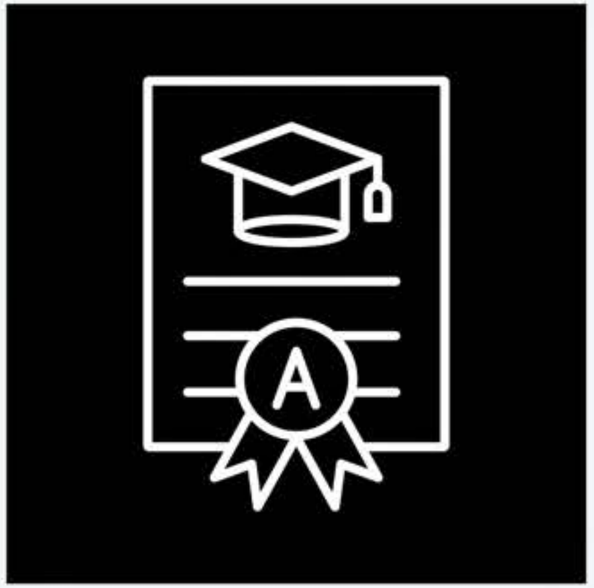
MODELS	Accuracy	Precision	Recall	F1 score
Logistic Regression	84.76%	73.61% | 73.91%	84.5% | 84.76%	78.1% | 78.43%
Naive Baye's classifier	93.9%	94.56% | 94.27%	94.19% | 94%	92.17% | 91.83%
Random forest	97.91	94.63% | 99.9%	93.77% | 99.9%	91.40% | 99.97%

Model Evaluation - sklearn vs. pyspark.ml



Outcomes

- **Deliver an accurate classification model that automates the sorting of Reddit posts into their respective subreddits.**
- **Help moderators identify and move posts to the appropriate subreddits.**
- **Improve discoverability of other lesser known, but more apt subreddit, for a more personalized user experience.**



Thank you!

Any questions?