

Скам/Спам сэтгэгдэл танигч

С.Буян-Эрдэнэ

О.Нямбаяр

М.Ариунзаяа

Б.Эрдэнэ-Очир

2025 оны 12-р сарын 5

Энэхүү төслийн ажилд Монгол хэл дээр бичигдсэн Facebook сэтгэгдлийг ашиглан тухайн сэтгэгдэл spam эсэхийг ангилах машин сургалтын загвар боловсруулах бүх үе шатыг тусгав. Үүнд өгөгдөл цуглуулалт, урьдчилсан цэвэрлэгээ, шинжилгээний аргачлал, шинж тэмдгийн боловсруулалт, мөн сургасан моделуудын бүтэц, ажиллах зарчим болон гүйцэтгэлийн үнэлгээ зэрэг гол хэсгүүд багтана.

Агуулга

1 Оршил	2
2 Өгөгдөл цуглуулалт	2
3 Өгөгдөл унших ба ачаалах процесс	3
3.1 Ачаалах үеийн алхмууд	3
4 Өгөгдөл цэвэрлэх ба бэлтгэх	3
4.1 Naive Bayes — текстэн өгөгдлийн цэвэрлэгээ	3
4.2 Decision Tree — тоон ба логик өгөгдөл бэлтгэх	4
5 Өгөгдөл хуваах	4
6 Машин сургалтын загварууд	4
6.1 Naive Bayes (Unigram & Bigram) модель	5
6.1.1 Сургалтын алхам	5
6.2 Decision Tree (Spam Feature Classifier)	5
6.2.1 Сургалтын алхам	6
6.3 Үнэлгээ хийх	6
6.3.1 Naive Bayes үнэлгээ:	6
6.3.2 Decision Tree үнэлгээ:	6
6.3.3 Үнэлгээний метрик	6
7 Өгөгдлийг визуализаци хийх	7
7.1 Сэтгэгдлийн урт (Comment length) тархалт	7
7.2 Emoji тоо	7
7.3 Үсгийн төрөл (Script types)	8
7.4 Binary features тархалт	8
7.5 Feature=1 үед тархалт	10
7.6 Correlation matrix	11
7.7 Өгөгдлийн тархалтын дүрслэлийн дүгнэлт	12
8 Консол application — SpamClassifierApp	13
8.1 Өгөгдөл ачаалах	13
8.2 Naive Bayes модель сургах	13

8.3	Decision Tree модель сургах	13
8.4	Модел үнэлэх	13
8.5	Өөрийн сэтгэгдэл ангилуулах	15
8.6	Визуализаци харах	16
8.7	N-gram vocabulary хадгалах	16
8.8	Гүйцэтгэсэн процесс	16
9	Дүгнэлт	17
10	Багийн гишүүдийн оролцоо	17
10.1	Хийх ажлын жагсаалт:	17
10.2	Хийсэн моделууд:	17
	Ашигласан материал	17

1 Оршил

Интернэт орчинд төөрөгдүүлсэн, сурталчилгааны зорилготой болон хуурамч мэдээлэл агуулсан spam сэтгэгдлүүд нь олон нийтийн мэдээллийн хэрэгслүүдийн аюулгүй байдлыг алдагдуулах томоохон асуудлын нэг юм. Тухайлбал, Facebook орчинд хийгддэг автомат сэтгэгдлийн спам нь хэрэглэгчийн туршлагыг муутгахын зэрэгцээ хуурамч зар сурталчилгааг түгээх, залилангийн эрсдэлийг нэмэгдүүлдэг. Энэхүү төслийн зорилго нь Facebook дээрх хэрэглэгчдийн бичсэн сэтгэгдлүүдийг Spam болон Ham гэж автоматаар ангилах явдал юм. Төслийн гол аргууд нь Decision Tree Learning [1] болон Naive Bayes Models [2] юм.

2 Өгөгдөл цуглуулалт

Өгөгдлийг Facebook-ийн олон нийтийн хуудас болон нийтлэлүүдийн доорх сэтгэгдлүүдээс гараар цуглуулж, Google Sheets дээр нэгтгэн хадгалсан. Сэтгэгдэл бүр дараах шинжүүдтэй:

- label - Spam эсэх (spam / ham)
- Комментарий бичсэн хүний нэр
- Raw comment - Тухайн бичсэн жинхэнэ текст

Цэвэрлэсэн сэтгэгдэл - Текстийг боловсруулсны дараах хувилбар

- Постны агуулга
- Зураг агуулсан эсэх
- Нэрээ нууцалсан эсэх
- Монгол нэр эсэх
- Голдуу ашигласан үсэг (Latin / Cyrillic)
- Emoji-ний тоо
- Сэтгэгдлийн урт
- Email/Link/Утас агуулсан эсэх

Сэтгэгдэл бүрийн хувьд текстэн болон тоон/логик шинжүүдийг бүрдүүлсэн ба эдгээр нь хоёр төрлийн загвар сургахад ашиглагдсан.

3 Өгөгдөл унших ба ачаалах процесс

Өгөгдлийг боловсруулахад зориулж **DataLoader** гэсэн класс бичсэн бөгөөд энэ нь:

- Google Sheets
- CSV файл
- Excel файл (.xlsx)

гэсэн гурван эх сурвалжаас ачаалах боломжтой.

Жишээ код:

```
df = pd.read_csv(url, engine='python', on_bad_lines='skip')
```

3.1 Ачаалах үеийн алхмууд

- Google Sheets линкээс CSV хэлбэрээр татаж уншсан
- Эхний мөрийг баганын нэр болгохоор шүүсэн
- SSL баталгаажуулалттай холбоотой алдааг түр хугацаанд алгассан
- **label** баганын утгуудыг: жижиг үсэг болгох, илүү хоосон зай арилгах, буруу өгөгдлүүдийг шүүх

Давуу тал :

- Олон төрлийн эх үүсвэрээс нэг ижил логигоор өгөгдөл ачаална
- `on_bad_lines='skip'` - буруу мөр илэрвэл код зогсохгүй
- Онлайн Google Sheets өгөгдлийг шууд уншина

4 Өгөгдөл цэвэрлэх ба бэлтгэх

4.1 Naïve Bayes — текстэн өгөгдлийн цэвэрлэгээ

`prepare_for_naive_bayes()` функц нь текстийг дараах 3 сонголтоор бэлтгэх боломжтой:

- **raw** — анхны бичсэн текст
- **transliterated** — кирилл рүү хөрвүүлж, цэвэрлэсэн текст
- **both** — raw + cleaned нэгтгэсэн текст

Цэвэрлэгээний үеийн алхмууд :

- Хоосон label/text мөрүүдийг хассан
- Цэвэрлэсэн текст байхгүй үед **fallback** хийдэг
- Эцэст нь сургалтанд бэлэн **x, y** Series буцаана:
 - `x` -> текст
 - `y` -> label

```
return df_clf['text'], df_clf['label']
```

4.2 Decision Tree — тоон ба логик өгөгдөл бэлтгэх

Decision Tree модел нь текст бус дараах төрлийн өгөгдлийг ашигладаг:

- Boolean (yes/no, true/false)
- Тоон үзүүлэлтүүд
- Label (spam / ham)

```
df[col] = df[col].map({
    True: 1, False: 0,
    "yes": 1, "no": 0,
    " ": 1, "   ": 0
}).fillna(0)
```

Boolean багануудыг 0/1 болгон хөрвүүлж, тоон багануудыг numeric болгож, label-ийг цэвэрлэсэн.

```
df[col] = pd.to_numeric(df[col], errors='coerce').fillna(0)
```

Label цэвэрлэгээ: - Spam -> spam - Ham -> ham

5 Өгөгдөл хуваах

Загварын гүйцэтгэлийг үнэлэхийн тулд өгөгдлийг: - **Train set — 80%**, - **Test set — 20%** хувиар хуваав. Хуваахдаа ангиллын тэнцвэрт байдлыг хадгалахын тулд **stratified split** ашигласан.

Naive Bayes-д зориулсан өгөгдөлд:

```
from sklearn.model_selection import train_test_split

X, y = data_loader.prepare_for_naive_bayes(text_source='raw')

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)
```

Decision Tree-д зориулсан өгөгдөлд:

```
df_features, attributes, label_col = data_loader.prepare_for_decision_tree()

train_df, test_df = train_test_split(
    df_features, test_size=0.2, stratify=df_features[label_col], random_state=42
)
```

DataLoader нь ML pipeline-д хэрэгтэй өгөгдлийг бүрэн бэлтгэж өгдөг.

Naive Bayes : - text - аbel - токенчлол хийхэд шууд бэлэн - lower-case - stopwords устгах боломжтой

Decision Tree : - Boolean шинжүүд - тоон үзүүлэлтүүд - label - attribute жагсаалт

Ингэснээр өгөгдөл Naive Bayes болон Decision Tree моделуудыг сургахад 100% бэлэн болсон.

6 Машин сургалтын загварууд

Энэхүү төсөлд Монгол хэлний спамыг илрүүлэх зорилгоор хоёр өөр алгоритмын загварыг харьцуулан судалсан. Текстэн өгөгдлийн онцлогоос хамааран Naive Bayes, тоон болон логик шинжүүдэд суурилсан Decision Tree алгоритмуудыг тохиромжтой гэж үзэн хэрэглэв.

6.1 Naive Bayes (Unigram & Bigram) модель

Төсөлд Multinomial Naive Bayes загварыг ашигласан. Энэхүү алгоритм нь бүх шинж чанарууд нь тухайн ангид нөхцөлт байдлаар хамааралгүй гэсэн хялбаршуулсан таамаглал дээр суурилдаг. Хэдийгээр энэ таамаглал бодит амьдрал дээр төдийлөн үнэн биш ч, Naive Bayes нь практикт, ялангуяа текстийн ангилалд гайхалтай сайн үр дүн үзүүлдэг [2].

N-gram токенчлол :

```
from naive_bayes_model import ngram_tokenize

text = "hello world nice day"
ngrams = ngram_tokenize(text, ngram_range=(1,2))
print(ngrams)
```

- Unigram: Нэг үг бүрийг тусад нь шинж болгон авсан
- Bigram: Хоёр үгийн дарааллыг нэмэлт контекст мэдээлэл болгон ашигласан
- ngram_range=(1,2): Unigram + Bigram токенчлолыг нэгэн зэрэг ашигласан

MyMultinomialNB класс :

- fit(X, y) → сургалт
- predict(X) → шинэ текстийг ангилах
- predict_single(text) → нэг текстийн ангилал
- save_vocabulary(filename) → vocabulary болон class тус бүрийн top n-grams-г файлд хадгалах

6.1.1 Сургалтын алхам

```
from naive_bayes_model import MyMultinomialNB
from sklearn.model_selection import train_test_split

# 1.
X, y = data_loader.prepare_for_naive_bayes(text_source='both')

# 2. /
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)

# 3. (Unigram+Bigram)
nb_model = MyMultinomialNB(alpha=1.0, ngram_range=(1,2))
nb_model.fit(X_train, y_train)

# 4.
y_pred = nb_model.predict(X_test)

# 5. Vocabulary-
nb_model.save_vocabulary("vocabulary.txt")
```

6.2 Decision Tree (Spam Feature Classifier)

Decision Tree нь текст бус логик болон тоон шинжүүд дээр суурилсан ангиллын модель юм. Decision tree сургалт нь индуктив таамаглалыг хэрэгжүүлэх хамгийн өргөн хэрэглэгддэг, практикт өндөр үр ашигтай аргуудын нэг юм. Энэ арга нь боломжит моддын орон зайд дээд-доош чиглэлтэй, greedy хайлтын стратеги ашиглан модоо бүтээдэг. [1]

Энгийн алгоритм (ID3) : - Энтропи (Entropy) тооцоолох - Information Gain тооцоолох - Багана тус бүрээр хуваалт хийх
- Рекурсив сурах

MyDecisionTree wrapper :

- `fit(df, attributes, target)` → Decision Tree сургах
- `predict(df)` → бүх dataset-д таамаглал хийх
- `predict_single(sample_dict)` → нэг дээжинд таамаглал
- `print_tree()` → модыг хүснэгт маягаар хэвлэх

6.2.1 Сургалтын алхам

```
from decision_tree_model import MyDecisionTree

# 1.
df, attributes, target = data_loader.prepare_for_decision_tree()

# 2.
tree_model = MyDecisionTree(max_depth=8)
tree_model.fit(df, attributes, target)

# 3.
tree_model.print_tree()

# 4.
y_pred = tree_model.predict(df)

# 5.
sample = df.iloc[0][attributes].to_dict()
print("Sample prediction:", tree_model.predict_single(sample))
```

6.3 Үнэлгээ хийх

ModelEvaluator класс нь дараах үйлдлүүдийг гүйцэтгэхэд зориулагдсан: - Сургалт: Хоёр загварыг тусад нь сургах - Үнэлгээ: Нарийвчлал, confusion matrix, precision/recall/F1-score тооцох - Таамаглал: Шинэ сэтгэгдэлд ангилал гаргах

6.3.1 Naive Bayes үнэлгээ:

```
#
,
model = MyMultinomialNB(alpha=1.0, ngram_range=(1,2))
model.fit(X_train, y_train)

#
y_pred = model.predict(X_test)
```

6.3.2 Decision Tree үнэлгээ:

```
#
,
tree_model = MyDecisionTree(max_depth=8)
tree_model.fit(train_df, attributes, target)

#
y_pred = tree_model.predict(test_df)
```

6.3.3 Үнэлгээний метрик

Моделийн гүйцэтгэлийг дараах метрикуудаар үнэлсэн: - Accuracy (Нарийвчлал): Нийт зөв таамаглалын хувь - Confusion Matrix: Анги тус бүрийн таамаглалын нарийвчлал - Precision (Нарийвчлал): таамаглагдсан spam-ийн эзлэх хувь -

Recall (Дамжуулалт): Жинхэнэ spam-ийн илрүүлэгдсэн хувь - F1-Score: Precision болон Recall-ийн дундаж

7 Өгөгдлийг визуализаци хийх

Visualizer нь: - Өгөгдлийг бэлдэх (`_prepare_data`) - Текст, emoji, binary болон numeric шинжүүдийг дүрслэх - Spam/Ham ангилалд хэрхэн тархсаныг харах

Дүрслэлүүд: - Сэтгэгдлийн урт (Comment length) тархалт - Emoji тоо - Үсгийн төрөл (Script types) - Binary features тархалт - Feature=1 үед Spam/Ham тархалт - Correlation matrix

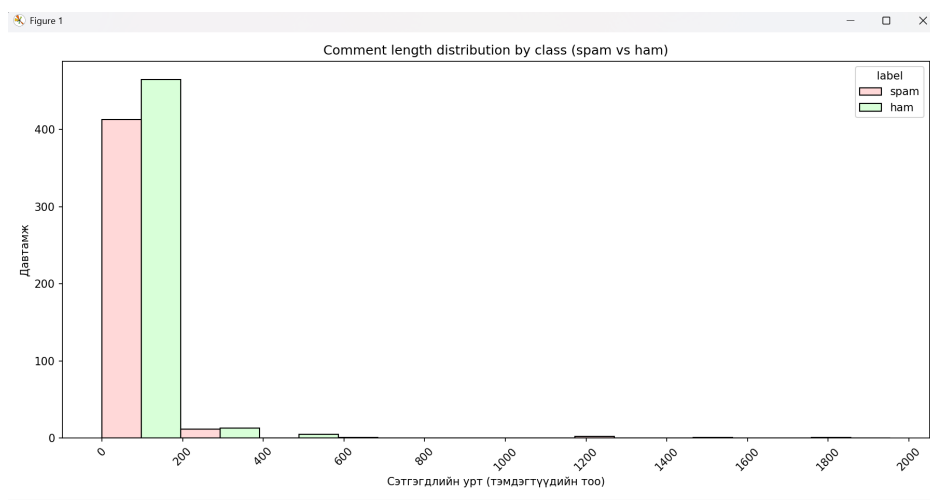
7.1 Сэтгэгдлийн урт (Comment length) тархалт

```
viz.plot_comment_length()
```

- Histogram дүрслэл: Spam vs Ham
- `bins=10, multiple='dodge'`
- X-тэмдэгтийн урт, Y-д давтамж

Үзүүлэлт: Spam сэтгэгдэл ихэвчлэн богино, Ham сэтгэгдэл харьцангуй урт

Зураг 1: Сэтгэгдлийн уртын тархалт

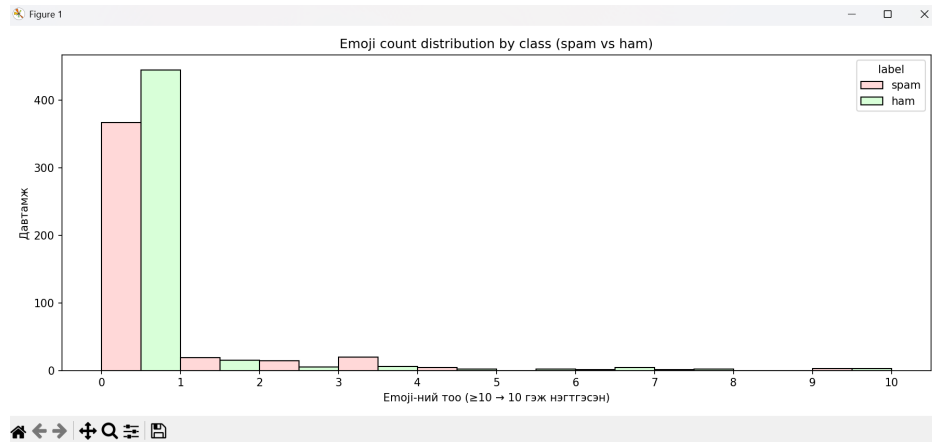


7.2 Emoji тоо

```
viz.plot_emoji_count()
```

- Emoji тоо ≤ 10 гэж нэгтгэж histogram үүсгэнэ
- Spam болон Ham-д хэрхэн тархсан харагдана

Зураг 2: Емоji тооны тархалт

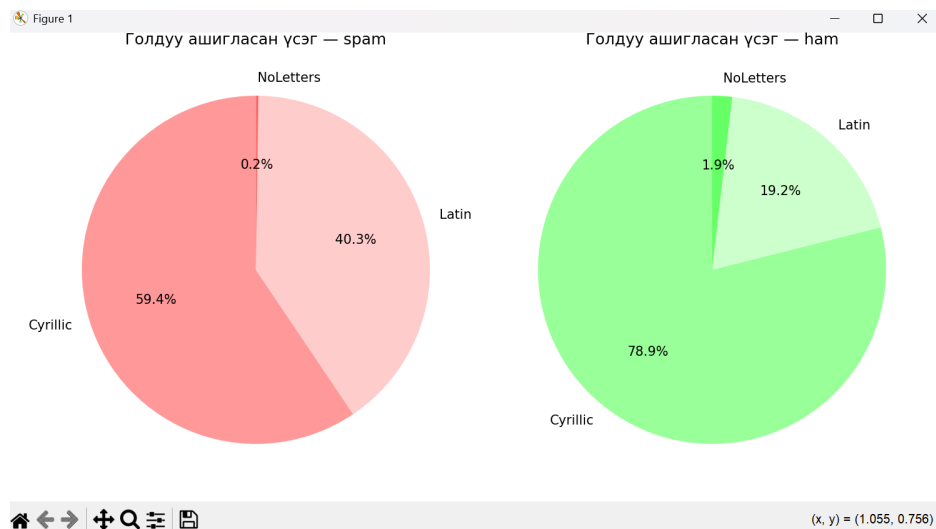


7.3 Үсгийн төрөл (Script types)

`viz.plot_script_types()`

- Pie chart: голдуу ашигласан үсгийн төрөл
- Spam/Ham тусад нь харуулна
- Script төрөл ба хувьсгалтай тархалт харагдана

Зураг 3: Script types



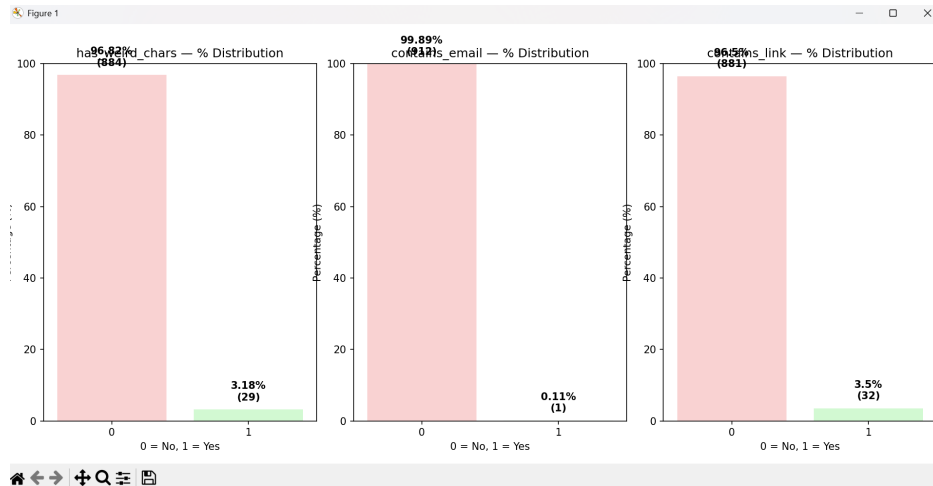
7.4 Binary features тархалт

`viz.plot_binary_distribution()`

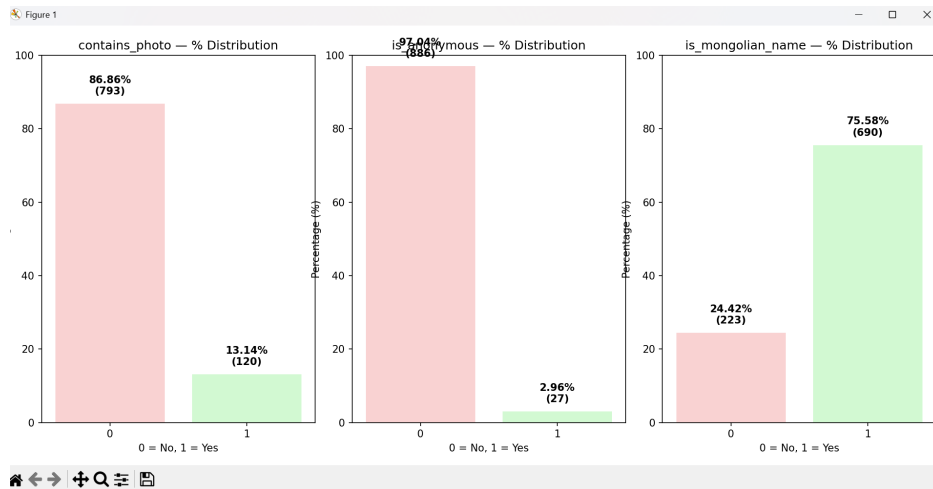
- Binary багануудын хувь (%) тархалт
- 0/1 утгуудын нийт хувь, count
- Spam/Ham аль нь илүү давамгай байгааг харах боломжтой

Binary features

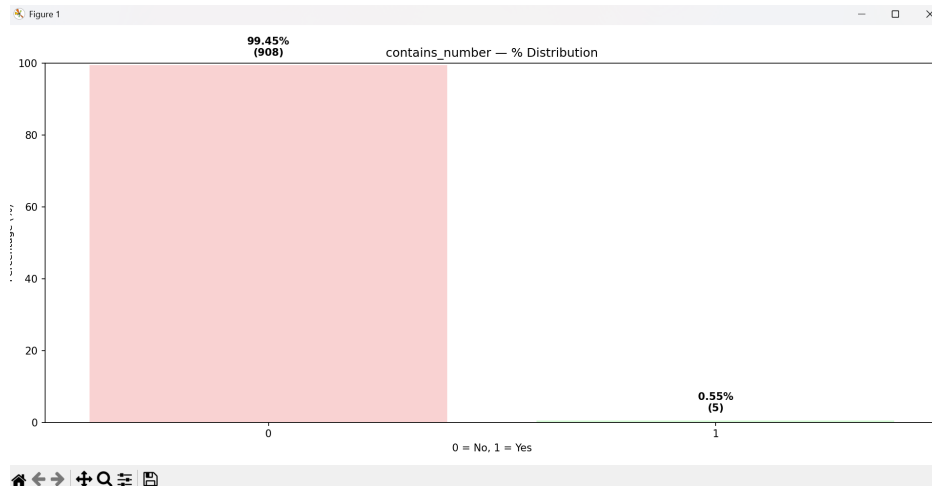
3ypar 4: feature1



3ypar 5: feature2



Зураг 6: feature3

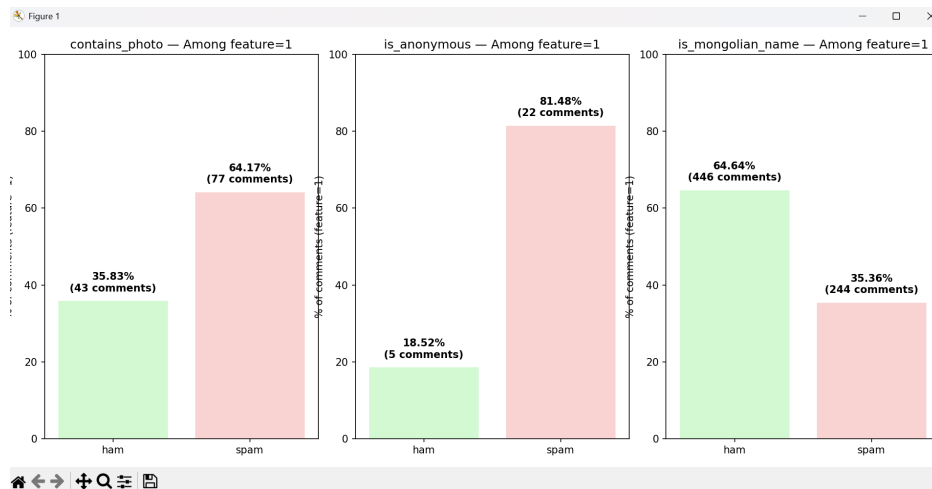


7.5 Feature=1 үед тархалт

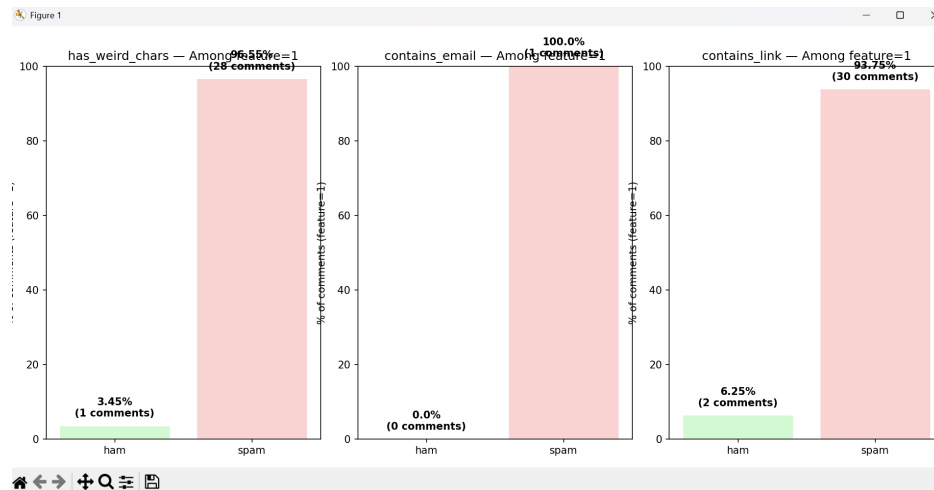
```
viz.plot_spam_ham_by_feature()
```

- Binary feature=1 үед Spam vs Ham хувь
- Жишээ: contains_link=1 → spam илүү давтамжтай

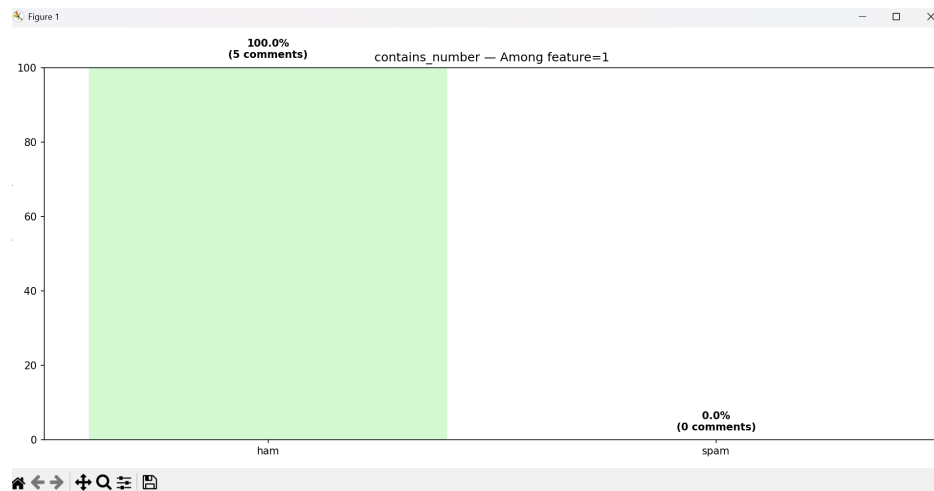
Зураг 7: feature1



Зүгар 8: feature2



Зүгар 9: feature3

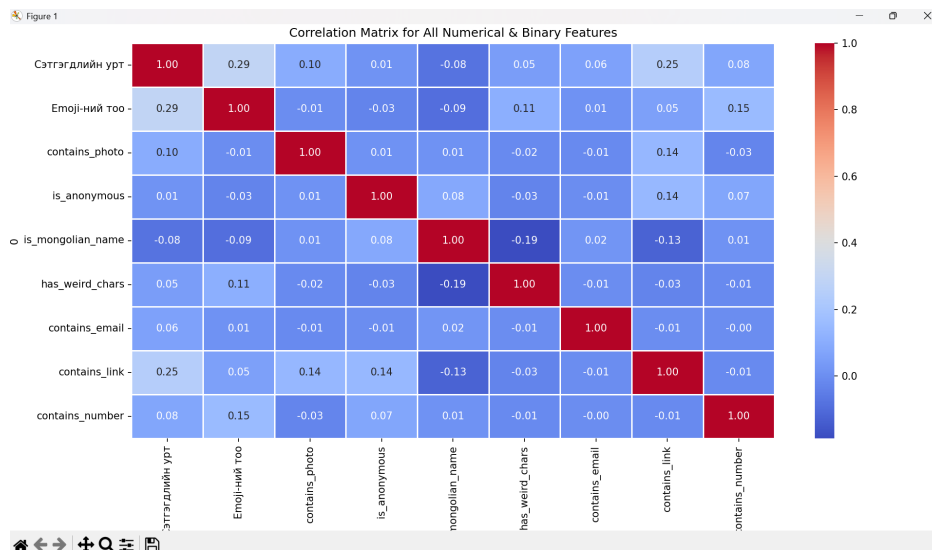


7.6 Correlation matrix

```
viz.plot_correlation_matrix()
```

- Тоон болон binary шинжүүдийн хамаарлыг heatmap-аар дүрслэнэ
- Feature хоорондын хамаарлыг 0–1 хүрээнд харах боломжтой

Зураг 10: Correlation Matrix



7.7 Өгөгдлийн тархалтын дүрслэлийн дүгнэлт

Visualizer нь өгөгдлийг цэвэрлэх, хувиргах, дүрслэх бүх алхмыг нэгтгэсэн бөгөөд Spam/Ham ангиллын тархалт, feature хувьсгал, correlation-ийг харах боломжтой. Үүнийг ашигласнаар текст ба binary feature анализ хийх, тархалтын судалгаа гаргах үйл явцыг хялбар болгоно.

Тархалтын графикууд нь spam болон normal сэтгэгдлүүдийн үгийн давтамж, урт, хамгийн түгээмэл үгсийн ялгааг ажиглахад тусалсан. Үүнд: - Сэтгэгдлийн урт

- Emoji-ийн тоо

- Голдуу ашигласан үсгийн төрөл (Latin / Cyrillic)

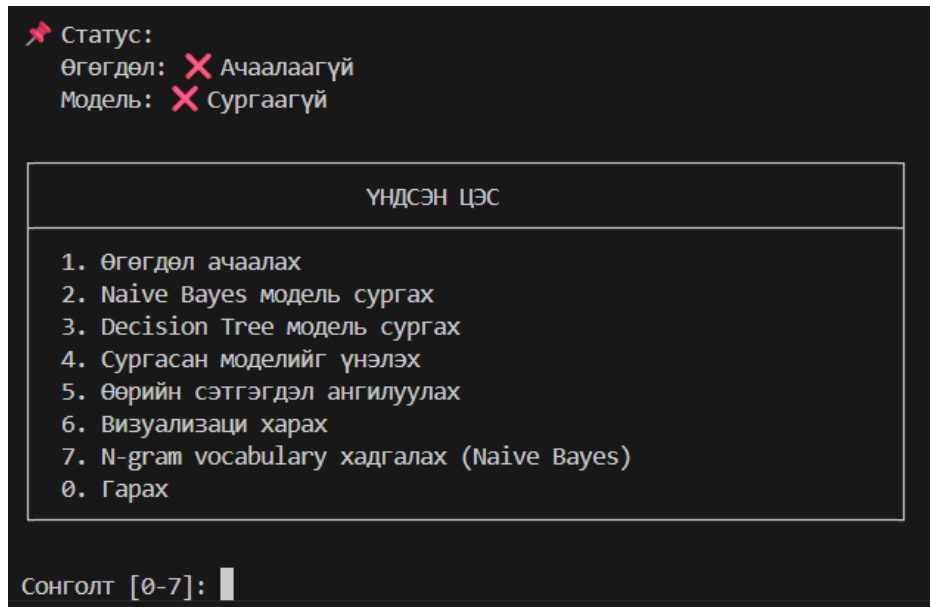
- Spam ангилалд хамгийн түгээмэл үгс

- Ham ангилалд хамгийн түгээмэл үгс багтана

Дүрслэлүүдээс харахад spam сэтгэгдэлд: - Spam сэтгэгдэлд мөнгө, зээл, чат гэх мэт үгс давамгайлсан. - Ham сэтгэгдэлд илүү энгийн, сэдвийн дагуу текст давамгайлсан. - Emoji болон бусад тусгай тэмдэгтүүд spam-д илүү их агуулагдсан байна.

8 Консол application — SpamClassifierApp

Зураг 11: Console App



SpamClassifierApp нь хэрэглэгчийн оролтод тулгуурлан дараах үйлдлүүдийг гүйцэтгэнэ:

8.1 Өгөгдөл ачаалах

- Google Sheets, CSV, XLSX файлаас өгөгдөл уншина.
- DataLoader ашиглан өгөгдлийг ачаална.
- Visualizer-д өгөгдлийг дамжуулж визуализацийг бэлдэнэ.
- Label тархалт, нийт мөрийг дэлгэцэнд харуулна.

8.2 Naive Bayes модель сургах

- Text эх үүсвэрийг сонгоно: raw, transliterated, эсвэл хоёулаа.
- Alpha (Laplace smoothing) болон n-gram төрлийг тохируулна.
- Test set-ийн хувь (default 30%) зааж өгнө.
- ModelEvaluator.train_naive_bayes дуудагдаж, модель сургана.
- Сургалтын болон test set-ийн мөрийн тоо, vocabulary хэмжээ хэвлэгдэнэ.
- Үүссэн n-gram vocabulary-г файлд хадгалах боломжтой.

8.3 Decision Tree модель сургах

- Maximum depth болон test set-ийн хувийг тохируулна.
- ModelEvaluator.train_decision_tree дуудагдаж, модель сургагдана.
- Сургалтын болон test set-ийн мөрийн тоо хэвлэгдэнэ.
- Хэрэглэгч хүсвэл tree бүтцийг console дээр харах боломжтой.

8.4 Модел үнэлэх

- ModelEvaluator.evaluate() дуудаж, сургасан моделийг үнэлнэ.
- Үүнд:

- Accuracy
- Confusion Matrix
- Class-wise Precision, Recall, F1-Score

Жишээ тайлан:

Зураг 12: Report Naive Bayes

```

МОДЕЛИЙН ҮНЭЛГЭЭ
=====
МОДЕЛИЙН ҮНЭЛГЭЭ
=====

Алгоритм: NAIVE_BAYES
Accuracy: 0.9377 (93.77%)

Confusion Matrix:
          Pred ham  Pred spam
Actual ham      143      13
Actual spam       4     113

Class бүрийн мэдээлэл:

ham:
  Precision: 0.9728
  Recall:    0.9167
  F1-Score:  0.9439

spam:
  Precision: 0.8968
  Recall:    0.9658
  F1-Score:  0.9300

=====
Дарж үргэлжлүүлэх...

```

Зураг 13: Report Decision Tree

```
SPAM/HAM CLASSIFICATION SYSTEM
Монгол хэлний сэтгэгдэл ангилагч

=====

МОДЕЛИЙН ҮНЭЛГЭЭ
-----

=====

МОДЕЛИЙН ҮНЭЛГЭЭ
=====

Алгоритм: DECISION_TREE
Accuracy: 0.6230 (62.30%)

Confusion Matrix:
          Pred ham  Pred spam
Actual ham      73      24
Actual spam     45      41

Class бүрийн мэдээлэл:

ham:
  Precision: 0.6186
  Recall:    0.7526
  F1-Score:  0.6791

spam:
  Precision: 0.6308
  Recall:    0.4767
  F1-Score:  0.5430

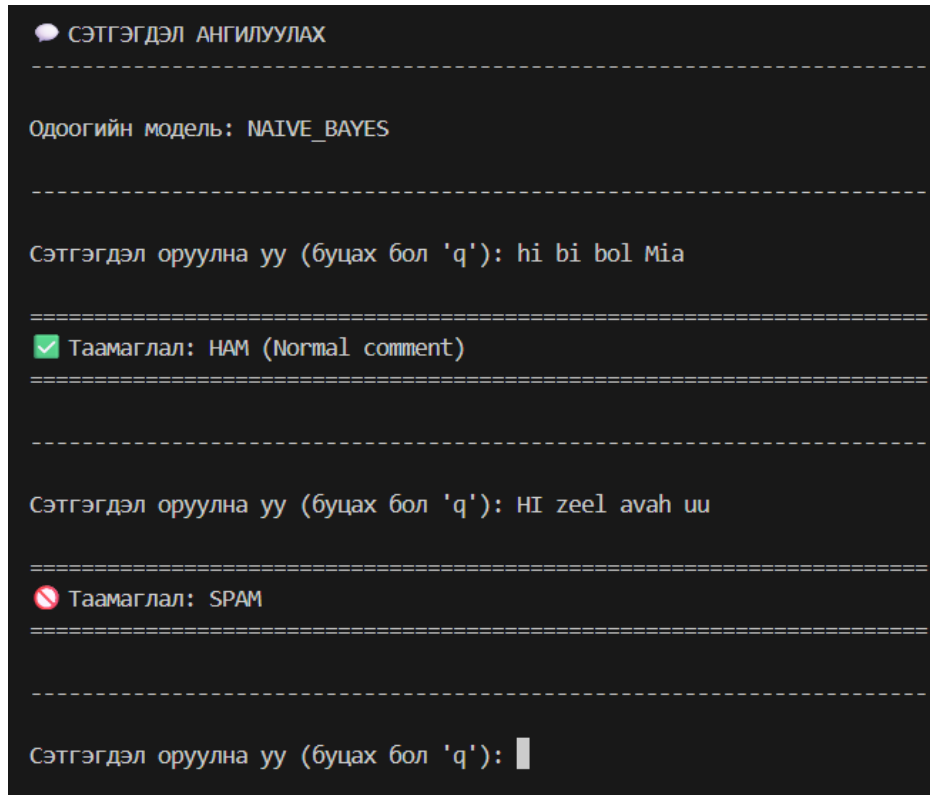
=====

Дарж үргэлжлүүлэх...
```

8.5 Өөрийн сэтгэгдэл ангилуулах

- Naive Bayes: Текст оруулахад spam/ham-г таамаглана.
- Decision Tree: Шинжүүдийг оруулахад spam/ham-г таамаглана.
- Үр дүнг тодорхой (SPAM, HAM) харуулна.

Зураг 14: Өөрийн сэтгэгдэл ангилуулах



8.6 Визуализаци харах

- Сэтгэгдлийн урт, emoji count, script types, binary feature тархалт, correlation matrix-г харуулна.
- Хэрэглэгч хүссэн графикаа сонгож эсвэл бүх график-г нэг дор үзэж болно.

8.7 N-gram vocabulary хадгалах

- Naive Bayes сурагдсан vocabulary-г текст файлд хадгална.
- Vocabulary хэмжээ болон файлын нэрийг дэлгэцэнд харуулна.

8.8 Гүйцэтгэсэн процесс

1. Өгөгдөл ачаална → DataFrame ба Visualizer бэлдэнэ
2. Модел сургана → Naive Bayes эсвэл Decision Tree
3. Модел үнэлнэ → Accuracy, Confusion Matrix, Precision/Recall/F1
4. Өөрийн сэтгэгдэл ангилна → Single prediction
5. Визуализаци хийж анализлана → Histogram, Pie chart, Heatmap
6. Vocabulary-г файлд хадгална (Naive Bayes)

9 Дүгнэлт

Энэхүү төсөл нь Facebook дэх Монгол хэлний спам сэтгэгдлийг илрүүлэх Decision Tree болон Naive Bayes загваруудыг харьцуулан судалсан. Үр дүнгээс үзэхэд текстэн шинжүүд (үгс, хэллэг) дээр суурилсан Naive Bayes загвар нь тоон шинжүүд (урт, эможи тоо) дээр суурилсан Decision Tree загвараас илүү үр дүнтэй, нарийвчлал сайтай ажиллаж байна. Энэ нь Монгол хэлний спамаг илрүүлэхэд тоон шинжүүд (урт, эможи тоо) нь туслах боловч текстэн өгөгдөл илүү чухал үүрэгтэй болохыг харуулж байна. Энэхүү Spam сэтгэгдэл илрүүлэгч жижиг систем нь одоогийн байдлаар зөвхөн зээлтэй холбоотой спам сэтгэгдлийг өндөр нарийвчлалтайгаар илрүүлэх боломжтой. Энэ нь хэрэглэгчдэд өдөр тутмын пост, коментуудаас зээлийн шинжтэй сэжигтэй мэдээллийг хурдан таньж, аюулгүй байдлаа хангахад үр дүнтэй шийдэл болж байна. Гэсэн хэдий ч бүх төрлийн спам (сурталчилгаа, залилан, фишинг, залилангийн холбоосууд гэх мэт)–ыг өндөр нарийвчлалтай илрүүлэхийн тулд илүү олон төрөл, олон эх сурвалжаас бүрдсэн том хэмжээний сургалтын өгөгдлийн сан шаардлагатай. Иймээс ирээдүйд системийг дараах чиглэлээр өргөжүүлэх боломжтой:

- Спамын төрөл бүрийн өгөгдлийг өргөн хүрээтэй цуглуулах
- Мэдээлэл боловсруулалтын нарийвчилсан аргачлал сайжруулах
- Илүү хөгжингүй ML/DL загвар ашиглан бүх төрлийн спамаг илрүүлэх чадварыг нэмэгдүүлэх

Эцэст нь, энэхүү төсөл нь анхан шатны түвшинд бодит хэрэглээнд ашиглагдах боломжтой, цаашид хөгжүүлбэл олон төрлийн spam-ийг илрүүлдэг бүрэн хэмжээний ухаалаг систем болох боломжтой гэж дүгнэж байна.

10 Багийн гишүүдийн оролцоо

10.1 Хийх ажлын жагсаалт:

- Өгөгдлөө цуглуулах (1 хүний 125 ham, 125 spam) : Бүгд
- Өгөгдлөө унших, цэвэрлэх, хуваах (CSV, excel format) : Нямбаяр
- Тархалтуудыг дүрслэх : Эрдэнэ-Очир, Нямбаяр
- PPT бэлдэх : Ариунзаяа
- Тайлан бичих (Quarto, qmd file) : Буян-Эрдэнэ

10.2 Хийсэн моделууд:

- Naive Bayes classifier (Unigram) - Буян-Эрдэнэ
- Naive Bayes classifier (Bigram) - Эрдэнэ-Очир
- Decision tree classifier - Ариунзаяа
- Unified model - Нямбаяр

Ашигласан материал

- [1] S. Russell and P. Norvig, “Decision tree learning,” in *Artificial intelligence: A modern approach*, 3rd ed., Prentice Hall, 2010, pp. 653–662.
- [2] S. Russell and P. Norvig, “Naive bayes models,” in *Artificial intelligence: A modern approach*, 3rd ed., Prentice Hall, 2010, pp. 758–760.