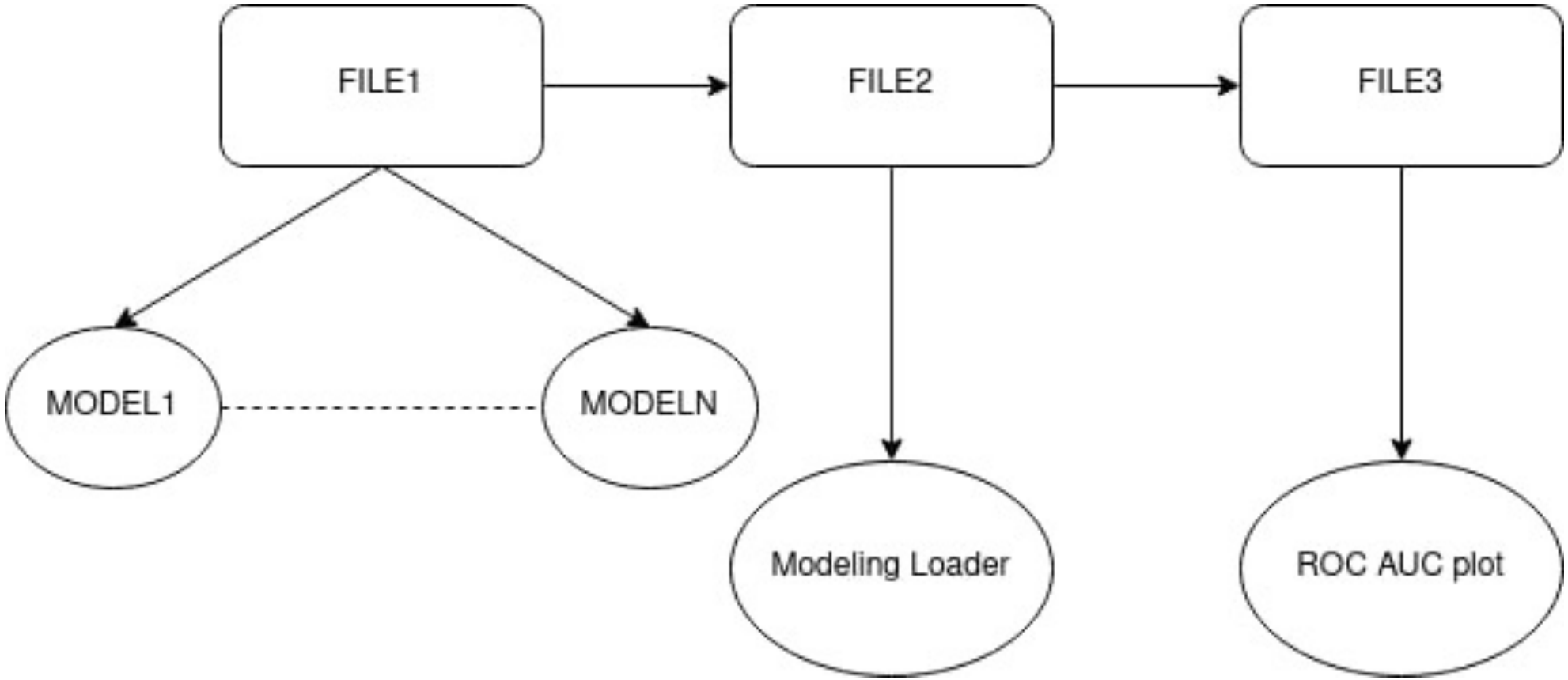
PO-sqrt-team

План работы

- Планирование
- Токенизирование предложения
- Создание моделей
- Тестирование разных моделей по файлу train.csv
- Построение пайплайна бэггинга над решающими деревьями
- Попытка обработки большого датасета
- Построение ROC AUC

Планирование



Токенизирование предложения

```
tokenizer = Tokenizer(num_words=5000, oov_token='<OOV>')
tokenizer.fit_on_texts(train_sentences)
word_index = tokenizer.word_index
train_sequences = tokenizer.texts_to_sequences(train_sentences)
train_padded = pad_sequences(train_sequences, maxlen=100, padding='post', truncating='post')
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(5000, 16, input_length=100),
    tf.keras.layers.GlobalAveragePooling1D(),
    tf.keras.layers.Dense(9, activation='relu'),
    tf.keras.layers.Dense(3, activation='softmax')
])
model.compile(
    loss='sparse_categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

Создание моделей

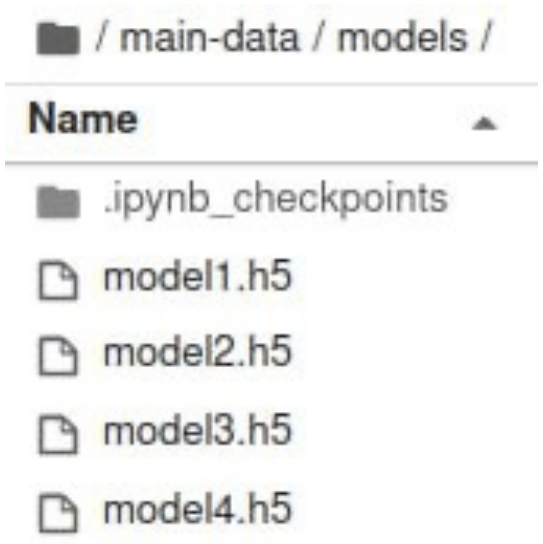
- 5000 -> 9*relu -> 3*softmax
- 5000 -> 9*tanh -> 3*sigmoid
- 5000 -> 9*exponential -> 3*softmax
- 5000 -> 9*relu -> 12*relu -> 3*softmax

На всех моделях функция потерь – sparce categorical crossentropy, оптимайзер – Adam, метрика – Accuracy, 15 epoch

Построение пайплайна бэггинга над решающими деревьями

Метод рассчета 3 параметров (‘-’, ‘+’, ‘?’) по методу бэггинга (ансамбль моделей):

1) Суммирование по каждому параметру
2) Деление каждого параметра на количество моделей в ансамбле




```
class ModelLoader():
    def __init__(self):
        self.models = []
        for i in range(1, 5):
            self.models.append(load_model('/home/jupyter/mnt/s3/main-data/models/model' + str(i) + '.h5'))

def predicted(sentence):
    res = [0.0, 0.0, 0.0]
    sequence = tokenizer.texts_to_sequences([sentence])
    padded = pad_sequences(sequence, maxlen=100, padding='post', truncating='post')
    predictions = loader.models[0].predict(padded)
    for i in range(3):
        res[i] += predictions[0][i]
    return res
```

Построение ROC AUC