Определение скоростей и размеров капель топлива при помощи машинного обучения

Выполнил:

Алешковский Александр Анатольевич

Техническое задание

- Сбор датасета из спектрограмм 4-х сигналов
- Вырезание окна сигнала, соответствующего капле.
- Определение параметров выреза и способа сглаживания данных
- Дополнение датасета расстоянием между пиками сигналов
- Дополнение датасета столбцами уже имеющихся признаков
- Расчет усредненных характеристик от сигналов
- Выделение геометрических признаков сигналов
- Применение базовой модели catboost для прогнозирования расстояния между пиками

Входные данные и их обработка


```python
def finder():

    path = os.getcwd()

    dir_pattern = re.compile(r"^M\d+$")
    file_pattern = re.compile(r"^M\d+_\d+\.mat$")

    # Список для собранных файлов
    collected_files = []

    for item in os.listdir(path):
        item_path = os.path.join(path, item)

        if os.path.isdir(item_path) and dir_pattern.match(item):
            for file in os.listdir(item_path):
                if file_pattern.match(file):
                    collected_files.append(os.path.join(item_path, file))

    collected_files.sort()
    print(collected_files)

    return collected_files
```

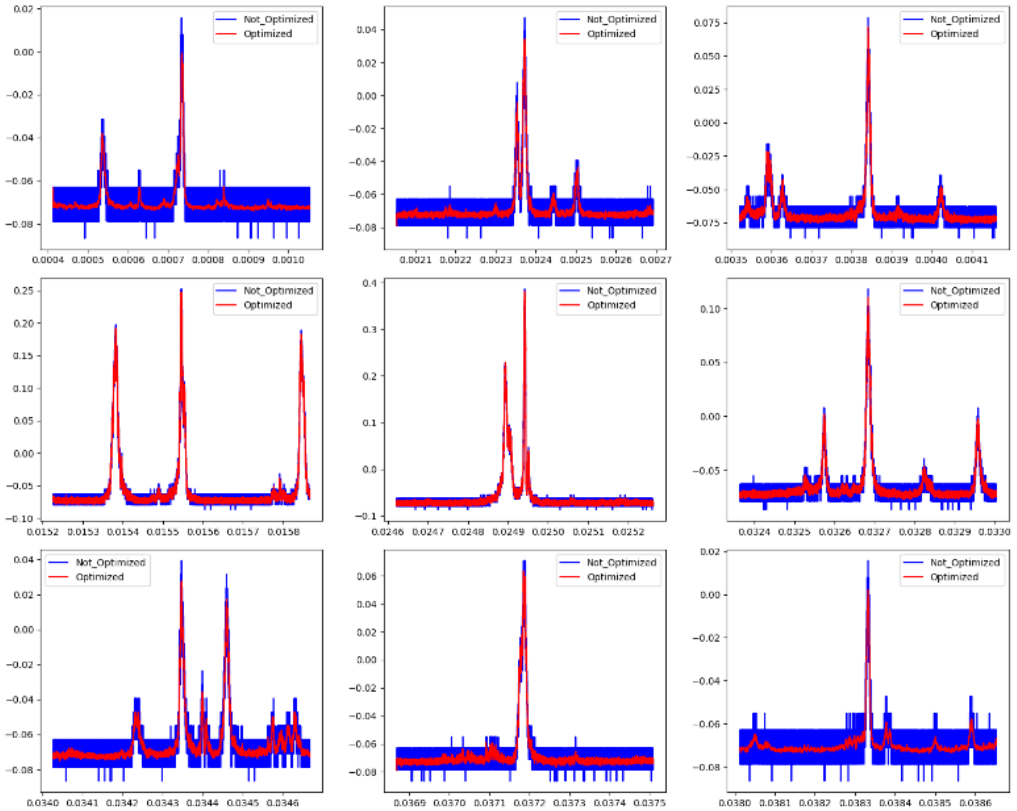
Входные данные представлены набором папок с названием «M{число}», где лежат файлы с названиями «M{Число}_{Число}.mat»

Файлы поочередно приводятся к формату pandas DataFrame и сглаживаются Low pass фильтром

```python
def apply_lowpass_filter(data, cutoff_freq=1.5, order=10, sample_rate=5):
    '''
    Применение фильтра к данным
    '''
    nyquist_freq = 0.5 * sample_rate
    normal_cutoff = cutoff_freq / nyquist_freq
    b, a = butter(order, normal_cutoff, btype='low', analog=False)
    filtered_data = filtfilt(b, a, data)
    return filtered_data
```

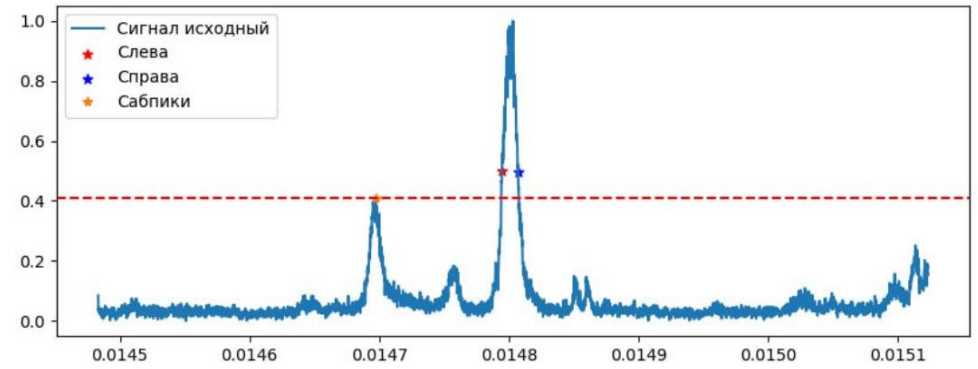
Low-pass filter (Фильтр низких частот) - это тип сигнального фильтра, который пропускает сигналы с частотами ниже определенной предела (называемого частотой среза), а блокирует те, которые находятся выше этого предела

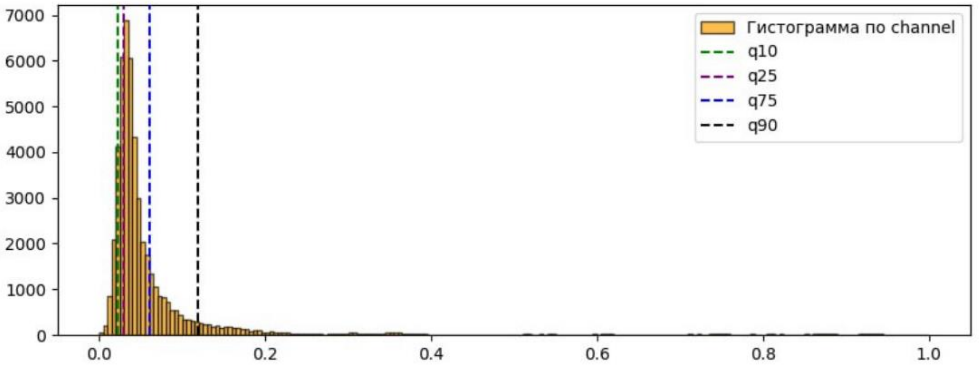
Сбор капель




```python
def raindrops_and_peaks(for_train, height_peak, window_size, butter=False):

    peaks, i = find_peaks(for_train['Channel A'], height=height_peak, distance=window_size)

    print(str(len(peaks)) + " length of peaks\n")

    # хранилище капель
    setup = np.empty((len(for_train),), dtype=object)
    print("Создано хранилище капель")

    # поиск пиков и вырез окна
    for i, peak in enumerate(peaks):

        start = max(peak - window_size // 2, 0)
        end = min(peak + window_size // 2, len(for_train))

        # Вырезаем окно для каждого канала
        window = for_train.loc[start:end, ['Time', 'Channel A', 'Channel B', 'Channel C', 'Channel D']].copy()

        # Добавляем окно в массив капелек
        if butter is True:

                for column in window.columns:

                    if column != "Time":

                        height_ = lambda a, b: a if a > b else b
                        window[column] = apply_lowpass_filter(window[column], height_(abs(window[column].max()), 0.0000000000001), 1, 2)

        window['ID'] = i

        setup[i] = window

    return setup
```

Извлечение признаков






```python
features = {
            'ID': ID,
            f'Minimum_{col_name[-1]}': channel_values.min(),
            f'Maximum_{col_name[-1]}': channel_values.max(),
            f'Variance_{col_name[-1]}': feature_calculators.variance(channel_values),
            f'Kurtosis_{col_name[-1]}': feature_calculators.kurtosis(channel_values),
            f'Skewness_{col_name[-1]}': feature_calculators.skewness(channel_values),
            f'Median_{col_name[-1]}': feature_calculators.median(channel_values),
            f'Std_{col_name[-1]}': np.std(channel_values),
            f'Semimax_{col_name[-1]}': geometricals.semi_max(channel_values),
            f'K_{col_name[-1]}': geometricals.k_semi_max(channel_values, time_values),
            f'Q10_{col_name[-1]}': geometricals.q10(channel_values),
            f'Subpeak_{col_name[-1]}': geometricals.subpeak(channel_values, time_values),
            f'Semiwidth_{col_name[-1]}': geometricals.semi_width(channel_values, time_values)[0] if isinstance(
                geometricals.semi_width(channel_values, time_values), tuple) else 0
        }
```

Извлечение признаков

- Minimum
- Maximum: Максимальное значение во временном ряду.
- Variance: Дисперсия временного ряда
- Kurtosis: Коэффициент эксцесса, измеряющий остроту (высокий или низкий) пика распределения.
- Skewness: Коэффициент асимметрии, измеряющий симметрию или асимметрию распределения.
- Median: Медиана временного ряда.
- Std: Стандартное отклонение временного ряда.
- Semimax: Полусумма максимальных значений (геометрический признак).
- K: Коэффициент K, связанный с положением и формой временного ряда (геометрический признак).
- Q10: Квантиль 10% временного ряда.
- Subpeak: Сумма амплитуд подпиков (геометрический признак).
- Semiwidth: Полуширина временного ряда (геометрический признак)

Настройка модели catboost

```python
def model_rain(features_df):

    columns = ['dtAB', 'dtCD', 'dtAC', 'dtBD']

    #Цели таргета
    y_train_multi = features_df[columns]

    #Исключение "лишних" обучающих данных
    X_train = features_df[[col for col in features_df.columns if col not in columns]]

    #Явное указание пула таргета и обучающих данных
    train_pool = Pool(data=X_train, label=y_train_multi)

    #Мультитаргетная модель
    model_multi = CatBoostRegressor(iterations=1000, depth=6, learning_rate=0.5,
loss_function='MultiRMSE',
                                    custom_metric=['MultiRMSE'])
    model_multi.fit(train_pool)
    predictions_multi = model_multi.predict(X_train)

    return predictions_multi
```

Руководство пользователя

1. Переместить папки с файлами в директорию с программой, или явно указать путь к файлам в коде программы. (по умолчанию – путь к папке с программой)
2. Ввести параметры «Размер окна для выреза» (по умолчанию – 50000), «Высота пика» (если не указано – выбирается автоматически с помощью квантиля)
3. Указать требуется ли создание графиков капель (по умолчанию – создается)
4. Указать параметры Low pass фильтра (если не указано – будет взято по умолчанию)
5. *Указать параметры модели CatBoostRegressor

*Напрямую в коде