

# Предсказание температуры во времени и пространстве

Дробин М.Е. (МФТИ ГУ)

*<2019-04-08 Пн>*

## Contents

<b>1</b>	<b>Введение</b>	<b>2</b>
<b>2</b>	<b>Обзор литературы</b>	<b>2</b>
<b>3</b>	<b>Получение данных</b>	<b>3</b>
<b>4</b>	<b>EDA</b>	<b>5</b>
<b>5</b>	<b>Эксперименты</b>	<b>6</b>
5.1	Схема валидации . . . . .	6
5.2	Baseline . . . . .	6
5.3	MLP . . . . .	7
5.4	LSTM . . . . .	10
5.5	SARIMA(facebook prophet) . . . . .	10
5.6	XGBoost . . . . .	13
5.7	Результаты экспериментов . . . . .	17
<b>6</b>	<b>Ссылки</b>	<b>17</b>

## Abstract

В этой работе исследуется качество прогнозирования температуры для одной из метеорологической станций, обучая нейросетевые алгоритмы(RBF, MLP) и классические(KNN, spatial averaging, inverse distance methods) по данным от других станций в пределах Англии.

## 1 Введение

Температура воздуха - временные ряды с высоким временным разрешением - измеряют только в немногочисленных, далеко разнесенных друг от друга метеорологических станциях. Поэтому появляется необходимость в пространственной интерполяции этих значений на области, где температура. Кроме пространственной интерполяции появляется необходимость в прогнозировании будущей температуры воздуха, т.е. временной интерполяции. Такие задачи появляются в сельском хозяйстве, прогнозировании погоды в городских условиях и т.д.

В качестве данных используется почасовая информация (влажность, ...) с 30 метеорологических станций Англии (каждая в отдельном городе) за 2 года. И для некоторых из них предсказывается температура. Данные взяты из сервиса darksky.

## 2 Обзор литературы

В статье Spatial Interpolation of Surface Air Temperatures... авторы сравнили качество MLP и spatial average, knn, inverse distance methods для задачи интерполяции температуры на 11 NOAA станций, на которых измеряют температуру. Эти станции расположены примерно в одном штате. Температура с этих станций - это множество ответов алгоритма, а данные с GCM (general circulation model) - численная модель - это множество объектов. Такая задача называется downscaling GCM. Трудность заключается в том, что пространственное разрешение выходных данных GCM - от  $2.5^\circ \times 2.5^\circ$  до  $8^\circ \times 10^\circ$  в долготу и ширину - это слишком грубо для предсказания - поэтому обучают по такой сетке нейросеть и предсказывают температуру для точек между сетки - выхода GCM. Авторы предсказывали максимальную температуру за день  $T_{\max}$  и посчитали качество нейросети (архитектура нейросети: 4-30-11 и 16-54-11 с сигмоидной функцией активацией). Для сетки с 4мя входами  $R^2$  и rmse были в среднем по 11 станциям 5.69 и 0.93; для сетки с 16ю входами - 5.12 и 0.94.

В статье Application of neural networks for the prediction of hourly mean surface temperatures in Saudi Arabia авторы использовали почасовые данные с шести метеорологических станций за 2 года с равнины, расположенной в Китае и ограниченной горами. 5 из них входят в множество объектов алгоритмов, для шестой предсказывается температура. 60% данных использовалось для обучения, остальное для теста. Сравнили MLP 5-17-1 с tanh активацией

и RBF с скрытым слоем из 7450 нейронов и гауссовой функцией активацией. Качество на тестовой выборке:  $R^2$  и  $\text{rmse}(^{\circ}\text{C})$  для MLP равны 0.96 и 1.067, а для RBF - 0.95 и 1.12 соответственно.

Данные по нескольким городам - погодные данные, взятые из измерителей в аэропортах этих городов. Задача - предсказать температуру(или прочие погодные показатели, напр., влажность..) для города, не использованного в обучении, используя нейросетевые алгоритмы.

### 3 Получение данных

Данные были получены через API сервиса darksky и библиотеки для Python darkskylib. Для каждого города были скачаны почасовые данные(иногда с пропусками) с 1 января 2009 по 1 января 2018 года. Следующие фичи были скачаны:

- температура воздуха(temperature)
- скорость ветра (windspeed)
- процент(от 0 до 1) покрытости неба облаками(cloudCover)
- относительная влажность, от 0 до 1 (humidity)
- видимость(visibility) - категориальная переменная, принимает только
- точка росы в градусах цельсия(dewPoint)
- краткое описание погоды - категориальная переменная (summary)
  - clear-day
  - clear-night
  - rain
  - snow
  - sleet
  - wind
  - fog
  - cloudy
  - partly-cloudy-day
  - partly-cloudy-night

- температура "по ощущениям" в градусах цельсия (apparentTemperature)

Алгоритм загрузки данных: for каждый город в списке городов: for каждая дата в списке дат с 2009 по 2018 год с периодом в 1 час: скачать исторические данные за эту дату для этого города добавить данные за этот город в общий датафрейм

```
cities = get_cities(['Oxford', 'Cambridge', 'Brighton And Hove', 'London'])
for city, key in zip(cities[: len(keys)], keys.keys()):
    df = pd.DataFrame()
    date_start = get_last_downloaded_date(city[0])
    date_end = dt(2018, 1, 1, hour=0)

    try:
date_list = get_list_of_days(date_start, date_end)
        except AssertionError:
continue

        logger.info(f"Скачиваю данные с города {city[0]}")
        for date in tqdm(date_list):
try:
            _city = forecast(key, city[1], city[2], time=date)
            error = False
except requests.exceptions.HTTPError as e:
            error = True
            logger.error(str(e.request) + str(e.response) + str(e))
            break
except Exception as e:
            error = True
            logger.error(str(e))
            break
try:
            for i in range(len(_city.hourly)):
values = [to_date_from_unix_time(_city.hourly[i]['time'])]
for column in columns:
                try:
values.append(_city.hourly[i][column])
                    except KeyError as e:
values.append(None)
```

```

t = pd.DataFrame(values).T

df = pd.concat((df ,t))
except AttributeError as e:
logger.error(str(e))
error =True

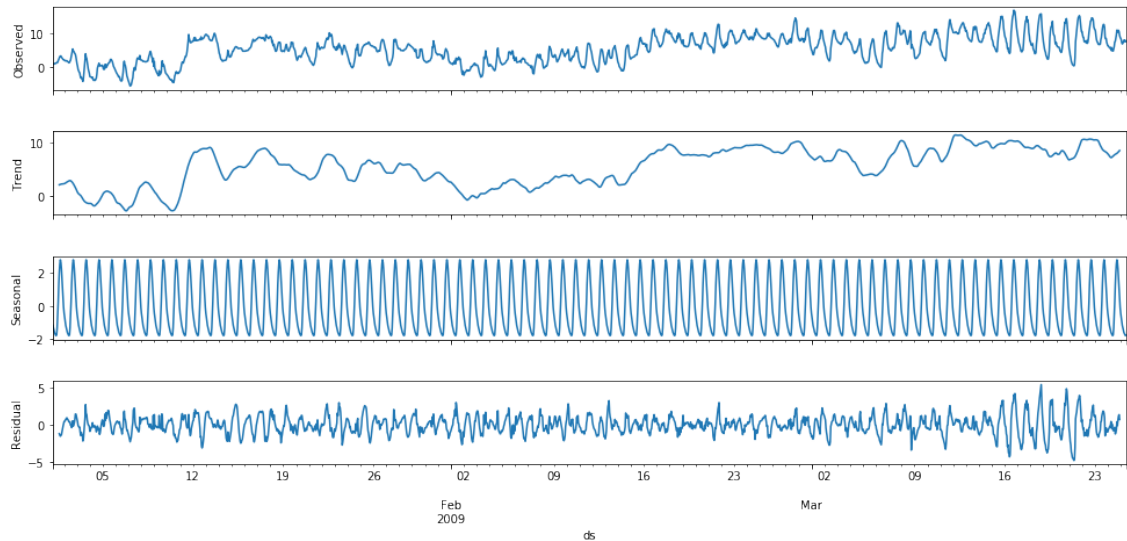
if df.shpape[0] > 0:
    df.columns = ["time"] + columns
    df = df.set_index("time")

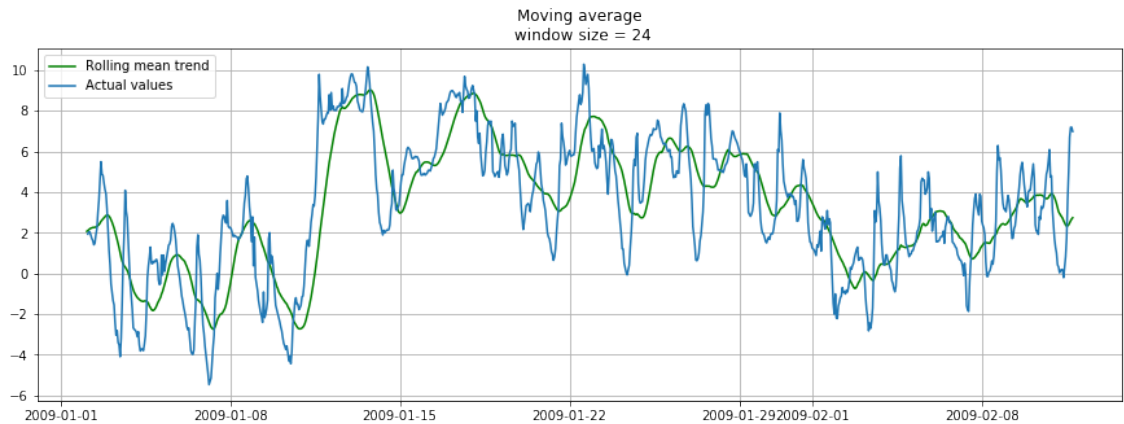
    path = os.path.join(CURRENT_DIR, "diplom_data/" + str(city[0]) + ".csv")
    with open(path, 'a') as f:
df.to_csv(f, index=True, header=False)

keys[key] = True #key is used, dont use it again today

```

## 4 EDA





- Есть ненулевой тренд
- четко выраженная дневная сезонность
- годовая сезонность

## 5 Эксперименты

### 5.1 Схема валидации

Модели обучались на первых 80% данных - до 2016-03-15. Валидировались модели на оставшихся 20% данных - около 2х лет. Причина выбора такой схемы валидации проста - у нас имеется относительно большое кол-во данных(в сравнии с чем?) и более сложные схемы валидации, например, cross-validation on a rolling basis, оказываются не нужны для построения устойчивой оценки алгоритма. Более сложные схемы валидации часто применяют, когда данных мало и делить исходную выборку на 2 невыгодно.

### 5.2 Baseline

В качестве алгоритма для сравнения было взято простое предсказание температуры, равное предыдущему значению:

```
def evaluate_naive_method():
    batch_maes = []
    for step in range(val_steps):
        samples, targets = next(val_gen)
        preds = samples[:, -1, 1]
```

```
mae = np.mean(np.abs(preds - targets))
batch_maes.append(mae)
return np.mean(batch_maes)
```

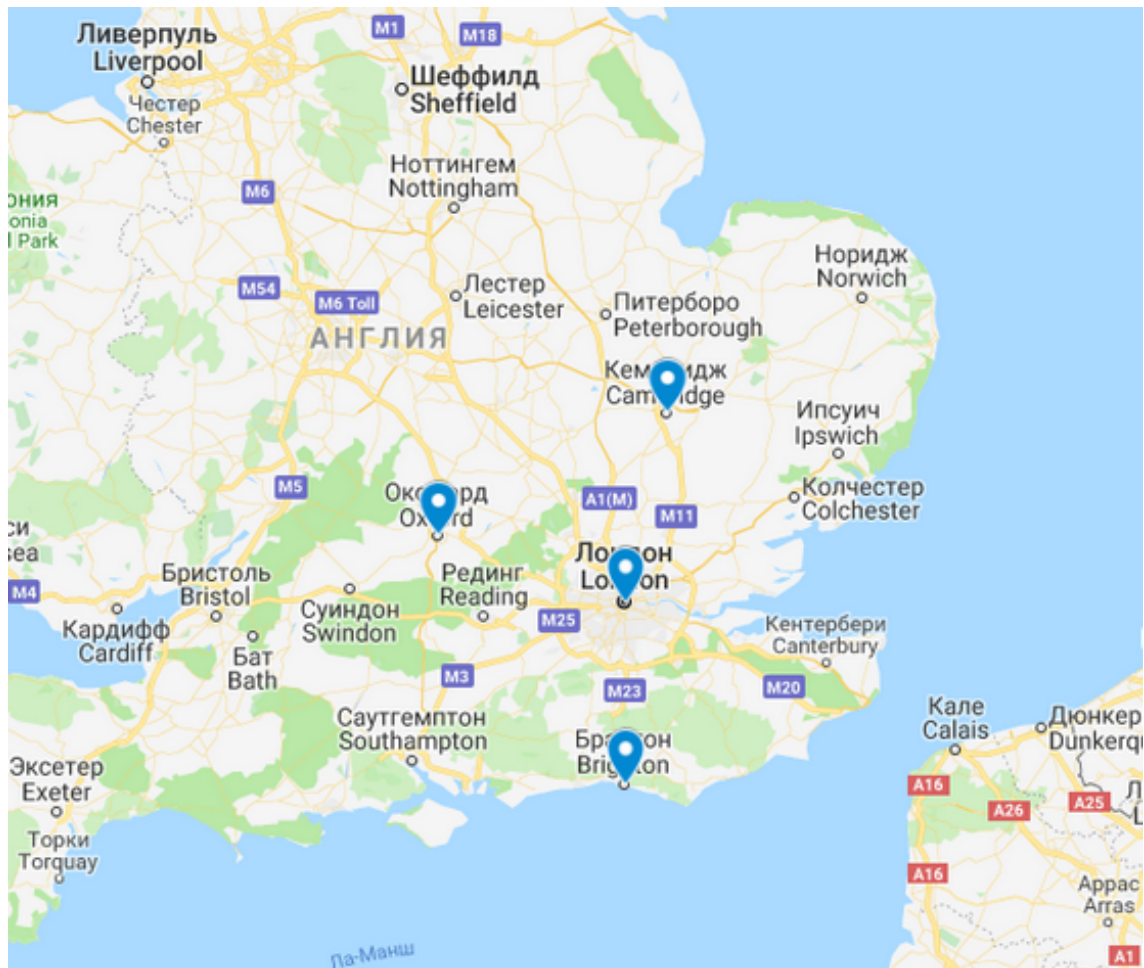
Этот алгоритм предсказывает на валидационной выборке с точностью до +/- 2.2 градуса Цельсия

### 5.3 MLP

Модель обучались на первых 80% данных - до 2016-03-15. Валидировалась - на оставшихся 20% данных - около 2х лет.

Данные были нормированы на среднее и дисперсию:  $x_i = \frac{x_i - \bar{x}}{\sigma}$ , где  $x$  - это отдельная фича или таргет (колонка в массиве объекты  $x$  фичи) и берется дисперсия и среднее этой фичи и она нормируется на свое среднее и свою дисперсию

фичи: только температуры 3 городов Оксфорд, Кембридж, Брайтон энд Хов, таргет - это Лондон.



MLP обучался следующим образом: брали данные за 5 дней и температуру Лондона на следующие 24 часа. Оптимизировали  $\text{mae}$ . Архитектура нейросети: полносвязный слой с 32 нейронами и  $\text{relu}$  активацией и полносвязный слой с одним нейроном без функции активации на выходе. Оптимизатор - RMSprop. Для более быстрой и лучшей сходимости, скорость обучения делилась на 10, когда функция потерь на валидации увеличивалась или не изменялась:

Генератор данных на керасе для обучения нейросети был заимствован из книги "Deep learning with Python". См. код здесь

```
model = Sequential()
model.add(layers.Flatten(input_shape=(lookback // step, data.shape[-1])))
```

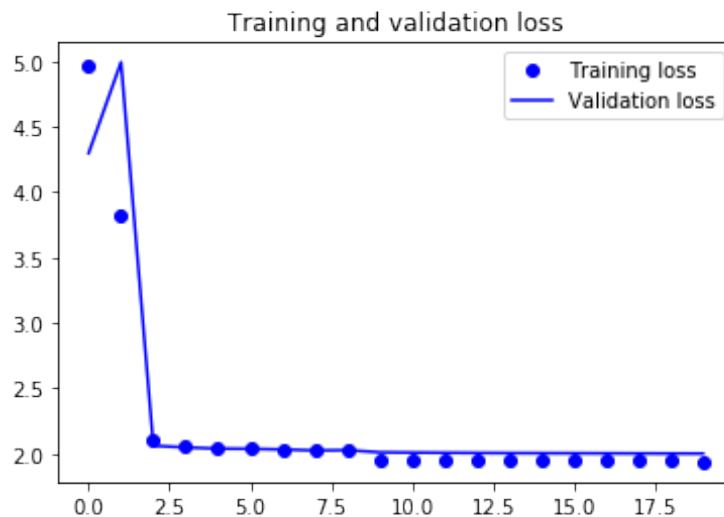


```

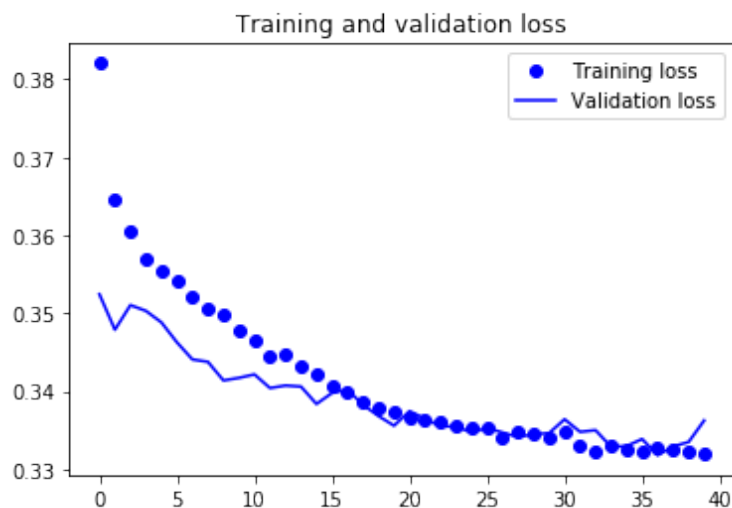
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dense(1))

model.compile(optimizer=RMSprop(), loss='mae')
callbacks_list = [
    keras.callbacks.ReduceLROnPlateau(
        monitor='val_loss',
        factor=0.1,
        patience=1,
        verbose = 1
    )
]
history = model.fit_generator(train_gen,
    steps_per_epoch=train_steps,
    epochs=20,
    validation_data=val_gen,
    validation_steps=val_steps,
    callbacks=callbacks_list)

```



Отсюда видно, что нейросеть выучила всю информацию из данных и строить модель сильнее нет смысла. Например, если попробовать обучить LSTM на тех же данных, то можно увидеть, что сеть не сможет превзойти результат MLP:



## 5.4 LSTM

LSTM обучается абсолютно так же, как и MLP

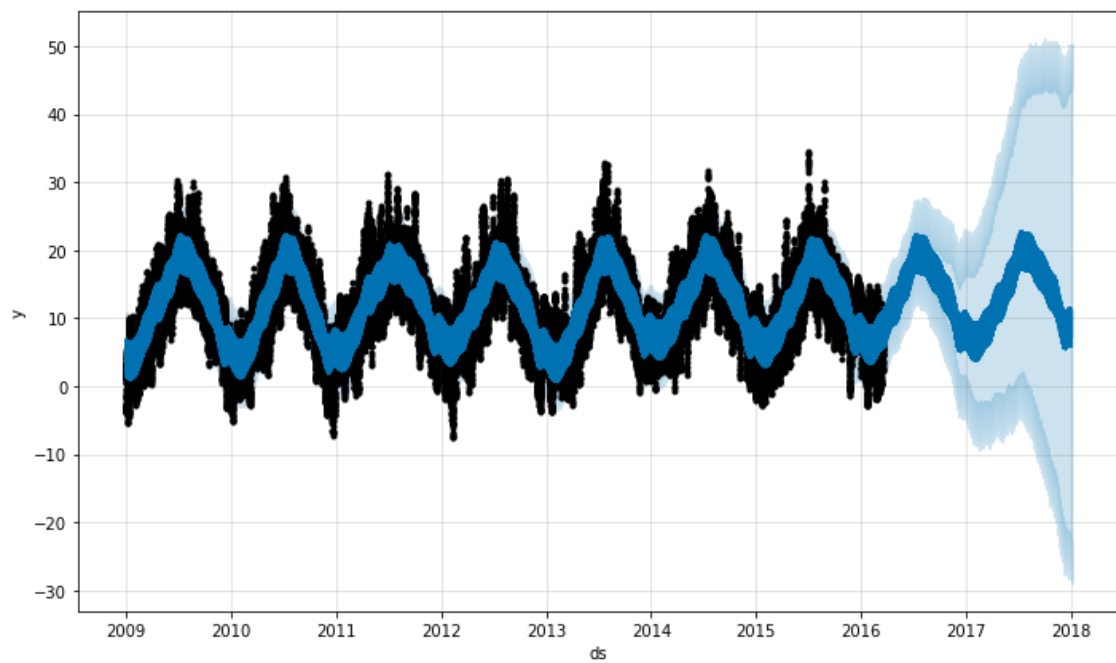
## 5.5 SARIMA(facebook prophet)

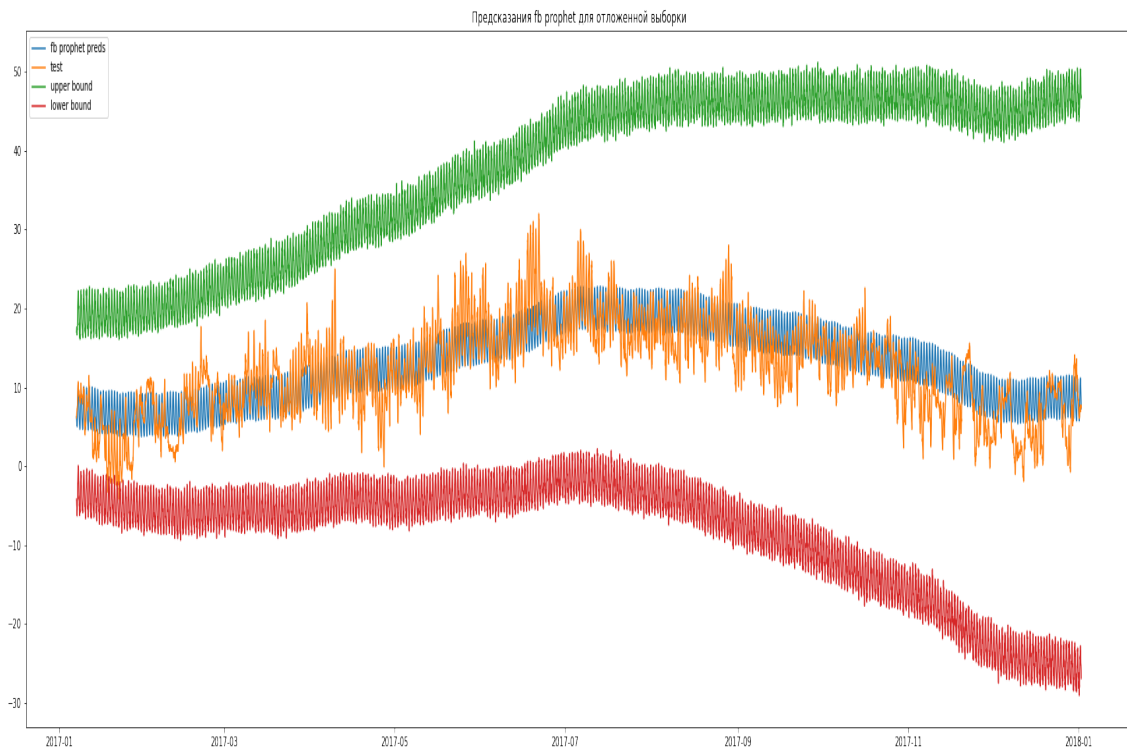
Так как наши данные периодичны с периодом в год, то вместо ARIMA, нужно использовать ARIMA с поддержкой сезонности - SARMIMA. SARIMA - это модель, которая обобщает линейную регрессию, взвешенное усреднение, дифференцирование временного ряда, экспоненциальное сглаживание. Это все простые модели, которые можно проверить на наших данных, используя только 1 модель - SARIMA.

SARIMA делает следующие допущения насчет данных - временной ряд стационарен:

- нет тренда
- нет сезонности
- дисперсия всюду одинакова

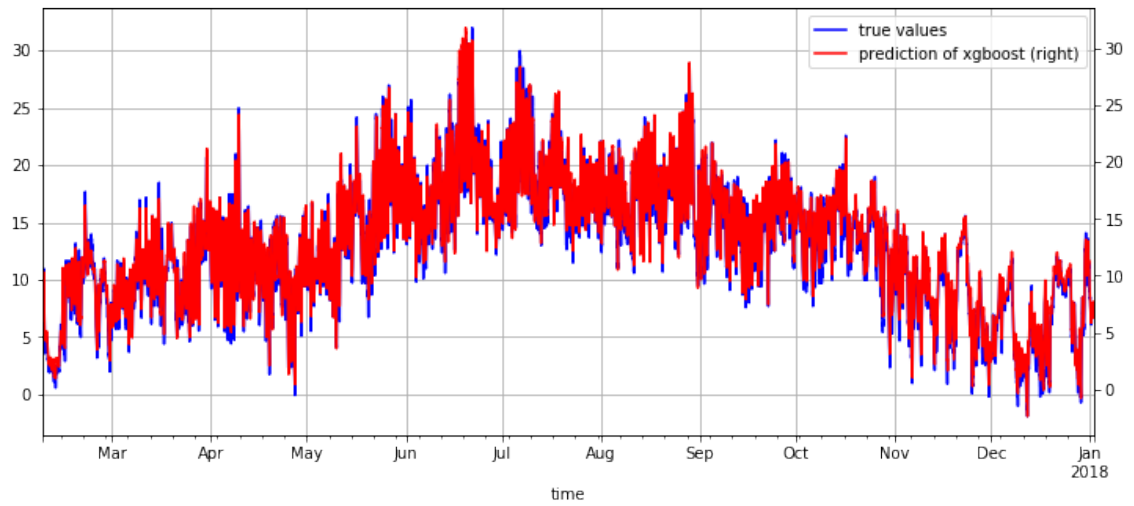
Проверку всех этих предположений, исправление нестационарного ряда в стационарный и применение SARMIMA реализовано в пакете facebook prophet

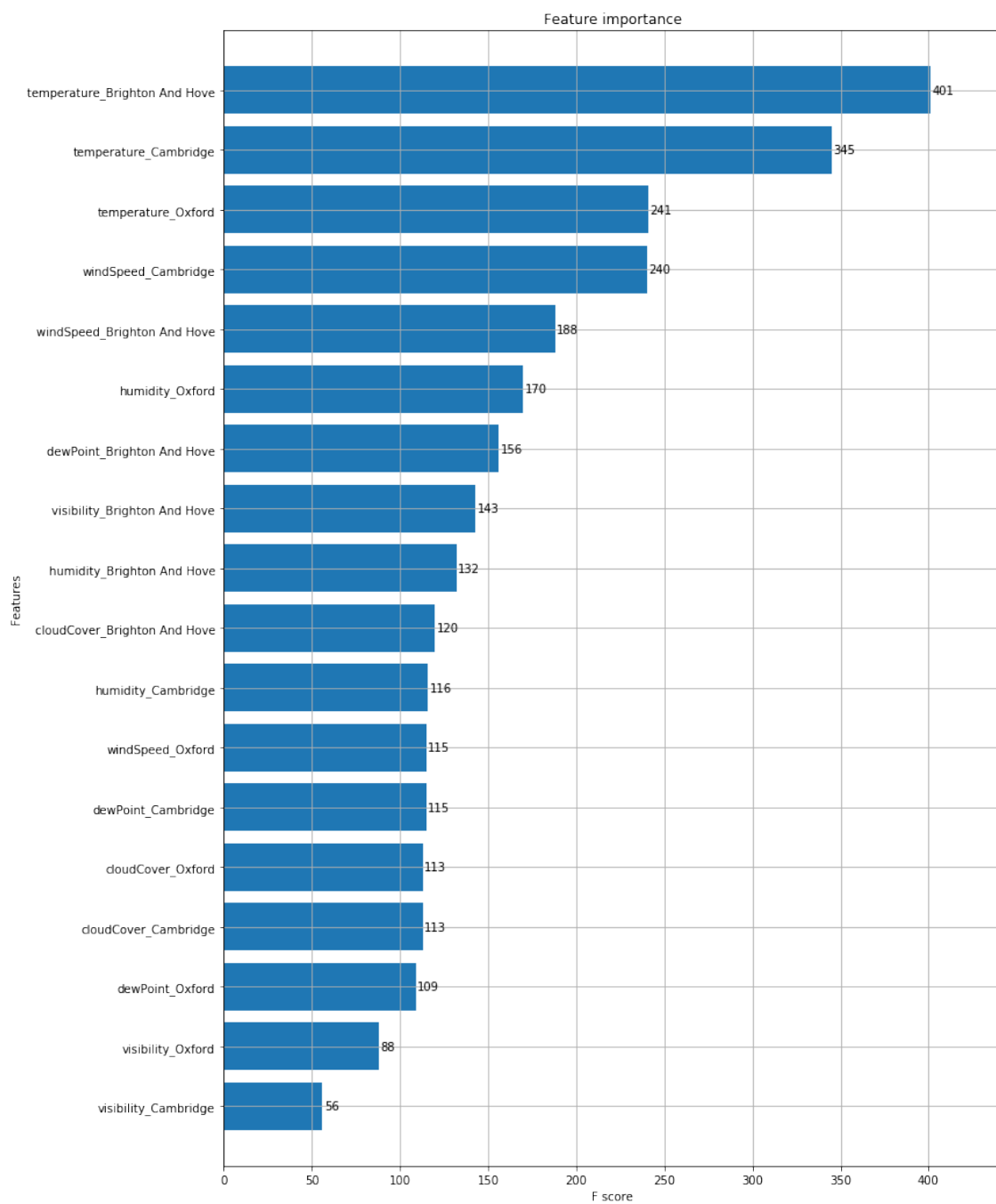




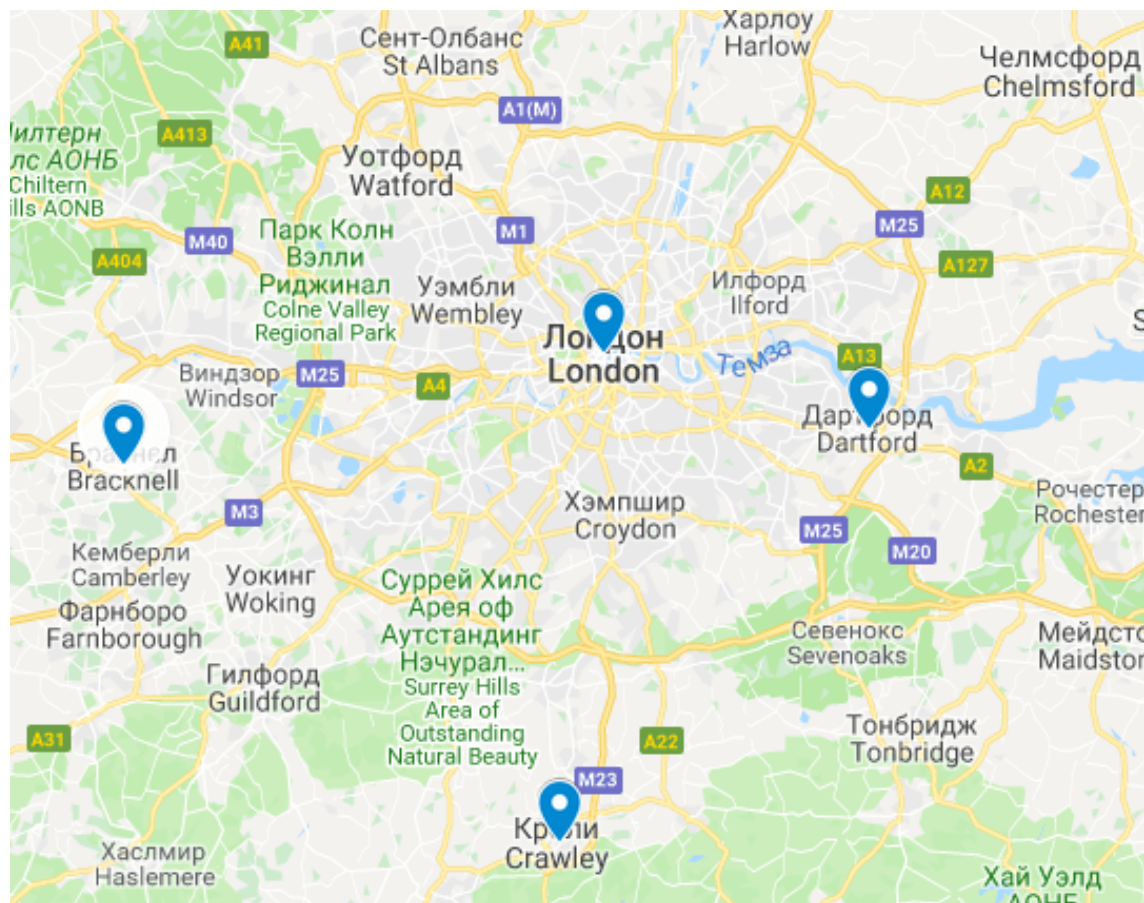
Из графиков видно, что fb prophet настраивается на тренд, но на колебания возле тренда настроится не может

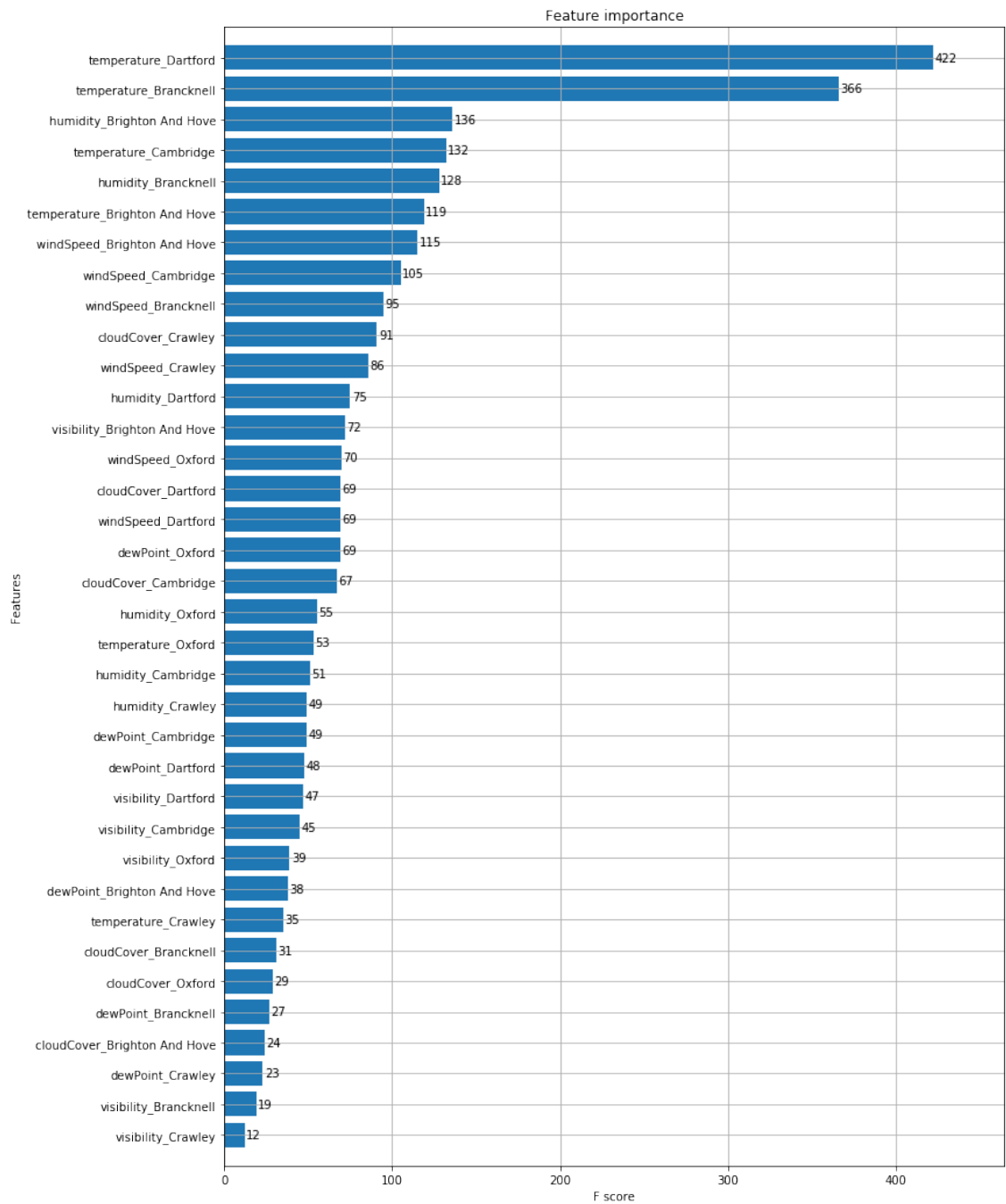
## 5.6 XGBoost



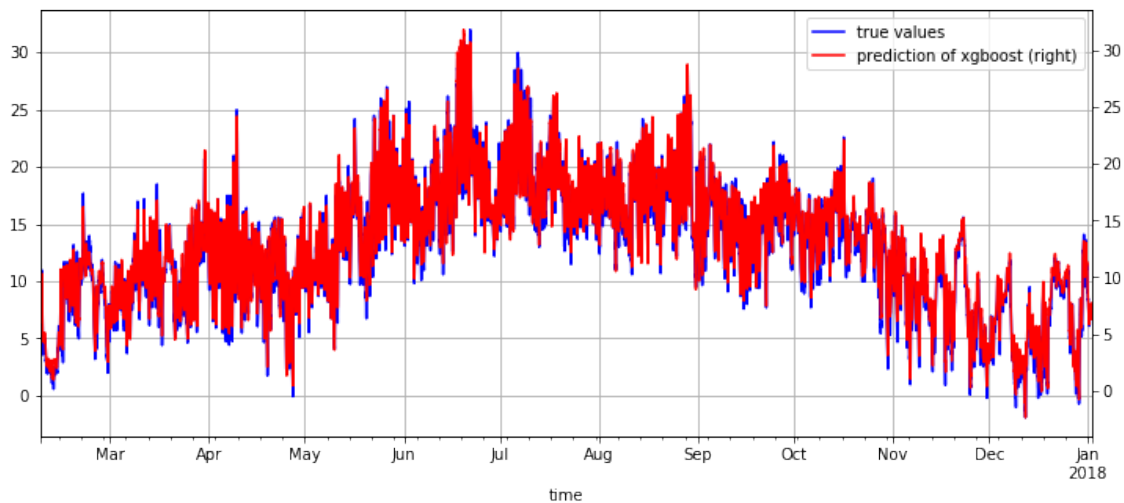


После добавления еще 3х городов между Брайтон энд Хов и Лондоном









## 5.7 Результаты экспериментов

baseline модель предсказывает температуру на следующий час по предыдущему значению, для нее нет смысла в разделении выборки на обучающую и тренировочную

модель	mae на валидационной выборке, градусы цельсия	mae на обучающей вы
baseline	2.20	-
MLP	2.00	
LSTM	1.99	
XGBoost, 6 городов	0.18	0.176
XGBoost, 3 города	0.46	0.427
SARIMA(fb prophet)	9.2	

## 6 Ссылки

Основные понятия и обозначения в машинном обучении. Воронцов К.В.