Rozdział 1

Wstęp

1.1 Opis projektu

Głównym celem projektu jest klasyfikacja urządzeń domowych na podstawie poboru prądu z użyciem ukrytych modeli Markowa. W projekcie korzystać będziemy z danych treningowych z pliku *house3_ devices_ train.csv*, w którym podany jest pobór prądu (tzw. "active power") zmierzony dla 5 urządzeń w odstępach około 20 sekundowych. Dane były mierzone w jednym domu w okresie od 16.04.2011, godz. 5:11:43 do 21.04.2011 godz. 7:21:44 (łącznie około 15000 pomiarów dla każdego urządzenia).

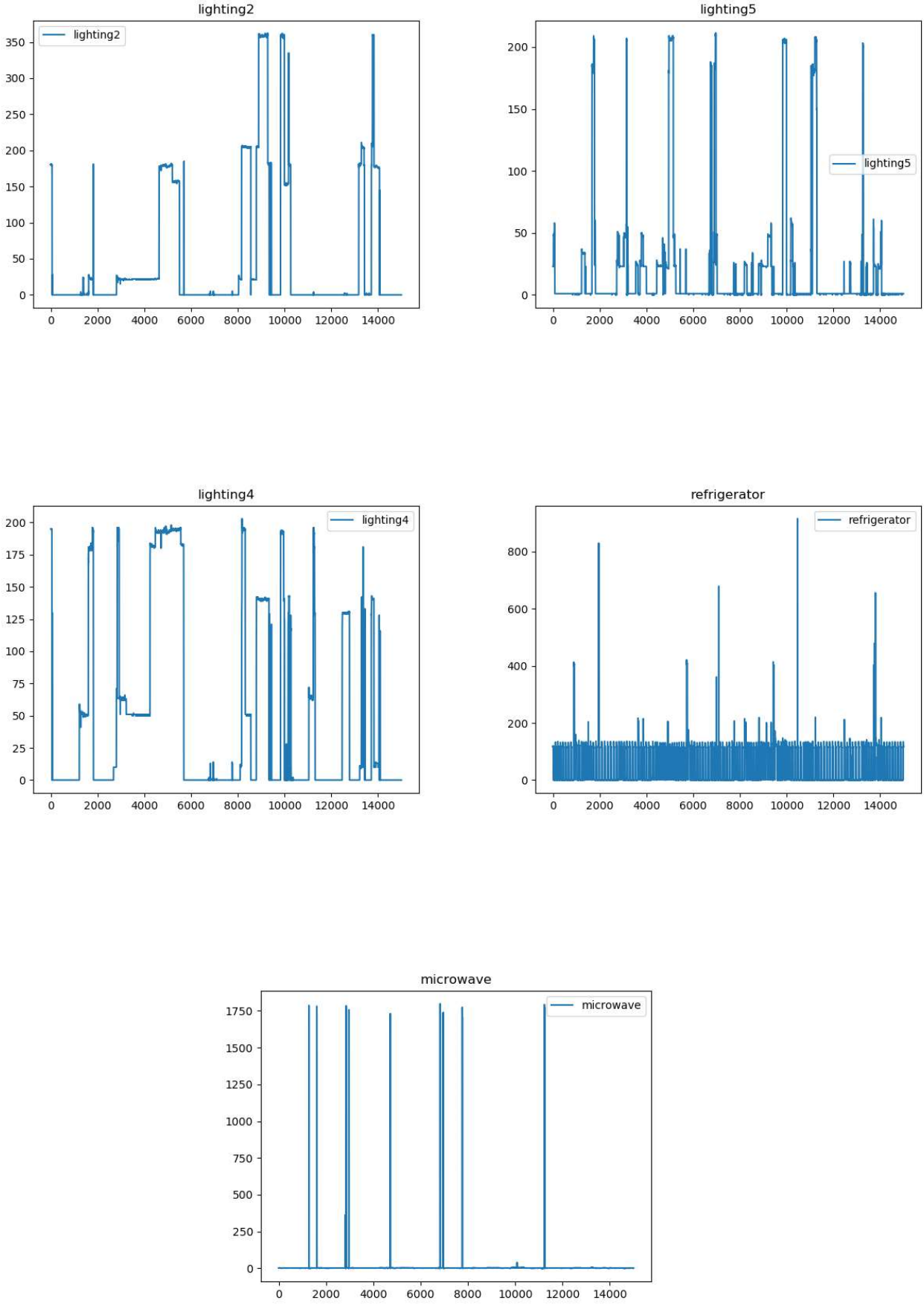
Dane, na których będziemy przeprowadzali badanie są fragmentem zbior *REDD* - link.

Plik ma następujący format:

```
1      time       lightiing2  lighting5  lighting4  refrgerator  microwave
2    1302930703     180          23         195          117          2
3    1302930721     181          23         195          119          2
4    1302930738     180          23         195          117          2
5    1302930765     181          23         195          117          2
```

Pierwsza kolumna to timestamp, z którego można odczytać dokładną datę oraz godzinę. W projekcie nie będziemy wykorzystywać tej wartości w procesie uczenia oraz procesie testowania. Wiersze traktujemy jako kolejne punkty czasowe, czyli kolejne momenty, w których dokonywane były pomiary. Kolejne 5 kolumn to pobór mocy dla każdego urządzenia. W omawianym zbiorze są informacje na temat 5 urządzeń: *lighting2*, *lighting5*, *lighting4*, *refrgerator*, *microwave*.

Wykresy poboru prądu na podstawie częściowego zbioru danych (*house3_ devices_ train.csv*) dla każdego urządzenia osobno wyglądają następująco:


lighting2
lighting2
lighting5
lighting5
lighting4
lighting4
refrigerator
refrigerator
microwave
microwave

Rysunek 1.1

1.2 Zadanie

Naszym zadaniem jest napisanie skryptu, który dla folderu testowego zawierającego pliki:

```
dev1.csv
dev2.csv
dev3.csv
...
```

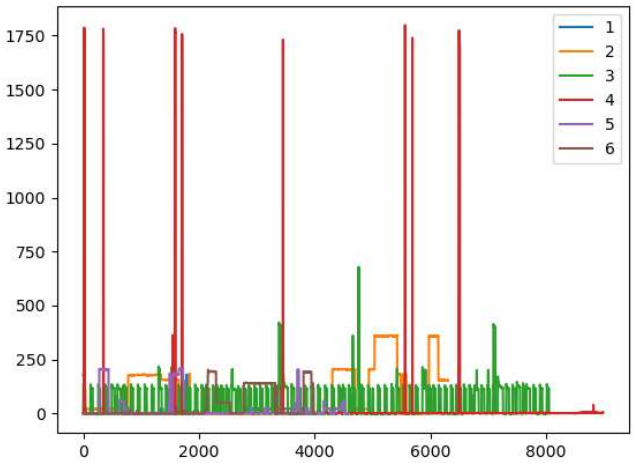
Z których każdy plik jest postaci:

```
    time      dev
1303001413     0
1303001430     0
1303001487    134
1303001509    132
1303001526    131
```

Jak najlepiej sklasyfikuje kolejne urządzenia z powyższego folderu, jako jedno z urządzeń opisanych wyżej: *lighting2*, *lighting5*, *lighting4*, *refrigerator*, *microwave*.

Widzimy, że każdy z plików *dev*.csv* jest tej samej postaci, czyli pierwsza kolumna to timestamp, którą ignorujemy w procesie testowania, natomiast druga kolumna to pobór prądu w kolejnych punktach czasowych. Warto wspomnieć, że każdy z następujących plików może być różnej długości oraz samych plików może być dowolnie dużo.

W naszym przypadku wszystkie testy przeprowadzane są na podstawie pliku załączonego na stronie z opisem projektu zawierającego pobór prądu dla sześciu urządzeń.
Wykres przedstawiający wszystkie urządzenia prezentuje się następująco:



Rysunek 1.2

Rozdział 2

Ukryte modele Markowa

Możemy zauważyć, że tak naprawdę naszym zadaniem jest klasyfikacja szeregów czasowych. W takim przypadku bardzo dobrym rozwiązaniem jest zastosowanie ukrytych modeli Markowa. W Pythonie jest zaimplementowanych kilka gotowych do tego bibliotek. W naszym projekcie korzystać będziemy z bilioteki ***hmmlearn***. Biblioteka hmmlearn implementuje ukryte modele Markowa (HMM) jako modele probabilistyczne, w którym sekwencja zmiennych obserwowanych X jest generowana przez sekwencję wewnętrznych stanów ukrytych Z.

Istnieją trzy podstawowe problemy związane z HMM:

- Biorąc pod uwagę parametry modelu i obserwowane dane, oszacowanie optymalnej sekwencji stanów ukrytych,
- Biorąc pod uwagę parametry modelu i obserwowane dane, obliczenie prawdopodobieństwa modelu,
- Biorąc pod uwagę tylko zaobserwowane dane, oszacowanie parametrów modelu.

Pierwszy oraz drugi problem można rozwiązać stosując algorytmy programowania dynamicznego znane odpowiednio jako algorytm Viterbi'ego oraz algorytm Forward-Backward.W implementacji drugiego z nich można użyć znanego nam już iteracyjnego algorytmu Expectation-Maximization (EM), inaczej nazywanego algorytmem Baum-Welcha.

W implementacji naszego projektu kilkukrotnie używamy funkcji ***GaussianHMM***, która (dla różnej liczby stanów ukrytych) buduje model HMM oraz za pomocą algorytmu Viterbiego pozwala zidentyfikować najbardziej prawdopodobną sekwencję stanów ukrytych na podstawie sekwencji obserwacji.

Następnie trenujemy nasze modele używając funkcji ***fit***. Biblioteka ***hmmlearn*** ma również zaimplementowaną metodę ***score***, która pozwala nam obliczyć logarytm funkcji wiarogodności dla modelu. Dzięki temu jesteśmy w stanie porównywać modele między sobą. Na kolejnej stronie opiszemy również inne kryteria, które umożliwiają nam dokonanie wyboru najlepszego modelu.

2.1 Kryteria wyboru modelu

Wybór najlepszego modelu dla każdego z pięciu urządzeń testujemy na podstawie następujących kryteriów:

- **logarytmu funkcji wiarogodności**,
- **AIC** (ang. *Akaike information criterion*),
- **BIC** (ang.*Bayesian information criterion*),
- **RIC** (ang. *Risk inflation criterion*).

Odpowiednie kryteria wyliczamy na podstawie poniższych wzorów:

$$\text{AIC} = -2logL + 2p,$$
$$\text{BIC} = -2logL + plog(T),$$
$$\text{RIC} = -2logL + 2plog(p-1),$$

gdzie p wyliczane jest na podstawie wzoru:

$$p = m^2 + km - 1,$$

gdzie:

- $logL$ to logarytm funkcji wiarogodności modelu,
- m to liczba ukrytych stanów w łańcuchu Markowa (u nas $m \in \{2, \ldots, 8\}$),
- k to liczba parametrów rozkładu (u nas k wynosi 2, gdyż obserwacje mają rozkład normalny oraz dla rozkładu normalnego estymujemy 2 parametry - średnią oraz wariancję).
- T to długość szeregu czasowego (liczba obserwacji dla danego urządzenia) .

W naszym projekcie testujemy każde z opisanych wyżej kryteriów i na podstawie wyników dla zbioru treningowego wybieramy najlepsze kryterium w docelowej implementacji. Z góry możemy założyć, że wyniki mogą nie być poprawne dla logarytmu funkcji wiarogodności, gdyż dla logarytmu funkcji wiarogodności często występuje zjawisko nadmiernego dopasowania do danych, czyli łatwo można model przetrenować. Natomiast kryterium BIC jest lepsze od kryterium AIC, gdyż ogranicza liczbę fałszywych odkryć, czyli popełnienie błędu I rodzaju.

Następnie, gdy mamy już wybrane pięć najlepszych modeli (jeden model dla każdego urządzenia) to dla każdego pliku ze zbioru testowego nakładamy po kolei te pięć modeli i liczymy dla nich logarytmy funkcji wiarogodności - tym samym uzyskując 5 wartości. Klasyfikujemy urządzenie jako to, dla którego logarytm funkcji wiarogodności osiąga największą wartość.

2.2 Klasyfikacja urządzeń dla kolejnych kryteriów

Ponownie przyglądając się wykresom poboru prądu dla kolejnych urządzeń ze zbioru treningowego (*Rysunek 1.1*) oraz wykresowi obrazującemu pobór prądu dla 6 testowych urządzeń (*Rysunek 1.2*) możemy najpierw wizualnie przyporządkować urządzenia, aby następnie zweryfikować otrzymane wyniki, dla kolejnych kryteriów.

Na podstawie wykresów możemy wyciągnąć następujące wnioski:

- tylko jedno z urządzeń możemy przyporządkować jako *microwave* (charakterystyczny wykres, na którym widzimy, że pobór prądu może osiągać wartość 1750), urządzenie z pliku 4,
- jedno z urządzeń możemy sklasyfikować jako *refrigerator* (ponownie na podstawie poborów w granicach 600-800), urządzenie 3,
- z pozostałych 4 urzadzeń tylko jedno ma zancząco wyższe wyniki (ponad 250, nie przekraczające tym samym wartości 400) od pozostałych, tak więc urządzenie 2 możemy sklasyfikować jako *lighting2*,
- urządzeń z poborami prądu z plików 1, 5 oraz 6 nie możemy jasno, jedynie na podstawie wykresów poprawnie przyporządkować, widoczną obserwacją jest jednak, że powinno być to jedno z urządzeń: *lighting5* lub *lighting4*.

Po analizie otrzymanych przez nas wyników, najlepsze pokrycie otrzymałyśmy dla algorytmu BIC, którego implementację postanowiłyśmy wykorzystać. Uzyskane w nim wyniki są postaci:

```
1     dev          dev
2 dev1.csv     lighting2
3 dev2.csv     lighting2
4 dev3.csv     refrigerator
5 dev4.csv     microwave
6 dev5.csv     lighting5
7 dev6.csv     lighting4
```

Pokrycie jest kompletne z domniemanymi wynikami, a więc wszystkie elementy zostały poprawnie sklasyfikowane.
W pliku *classify_ devs_ numerIndexu.py* zawaryłśmy implmentację każdego z kryteriów, z czego wszystkie poza BIC są zakomentowane (do wglądu).