Politechnika Wrocławska

Probablistyczne Modele Grafowe

Projekt: Predykcja pogody

Imię, nazwisko, numer albumu

Dariusz Palt, 246808

Eryk Wójcik, 259311

Bartłomiej Gintowt, 262225

Prowadzący

Mgr inż. Denis Janiak

Grupa zajęciowa

W04SZT-SM0021G, grupa 2

Termin oddania

23 czerwca 2024

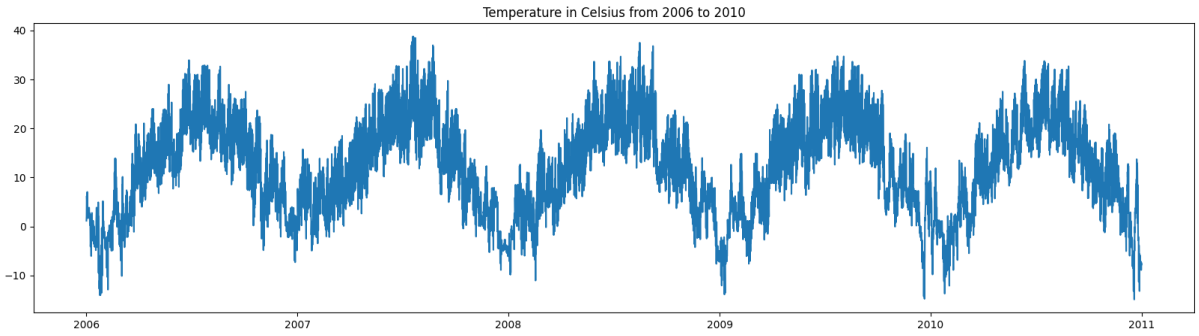
1 Cel ćwiczenia

W ramach projektu PGM zdecydowaliśmy się na podjęcie zadania predykcji pogody. Chcieliśmy porównać metody poznane na laboratorium z innymi, mającymi podobne zastosowania.

2 Eksploracyjna analiza danych

Przeprowadziliśmy eksploracyjną analizę danych zbioru **Weather Dataset**.
Kroki EDA:

1. Wczytanie zbioru danych.
2. Auto EDA (implementacja z $ydata\_profiling$), wnioski:
   - kolumna $LoudCover$ ma stałą wartość,
   - kolumna $Summary$ jest niezbalansowana,
   - wiele kolumn ma wartości zerowe,
   - nie ma brakujących danych.
3. Usuwamy kolumnę $LoudCover$, gdyż nie daje nam nowych informacji.
4. Wartości zerowe zostawiliśmy takie jakie są, przy danych pogodowych często mogą takie występować (na przykład kolumna $WindSpeed$ ma wartość 0 gdy nie ma wiatru).
5. Badamy korelację: zmienne które nie powinny nie są ze sobą skorelowane.
6. Sortujemy zbiór danych po czasie.
7. Próbkujemy pomiary temperatur co 6h, zamiast co godzinę.
8. Dodajemy kolumnę $Temperature[K]$ w stopniach Kelvina.
9. Dodajemy kolumny $Hour$ oraz $Month$.
10. Filtrujemy zbiór danych po czasie: lata 2006-2010.
11. Zapis zbioru danych do pliku .csv.



3 Modele

Zaimplementowaliśmy 3 modele:

- CRF, MLP + CRF, LSTM + CRF,
- ARIMA,
- BayesianLSTM.

Modele te wybraliśmy ze względu na to, że mogą pracować na danych sekwencyjnych i widzimy potencjał w ich wykorzystaniu do predykcji temperatury.

3.1 CRF, MLP + CRF, LSTM + CRF

3.1.1 CRF

CRF (Conditional Random Fields) to klasa metod modelowania statystycznego, często stosowana w rozpoznawaniu wzorców, używana do strukturalnego przewidywania. W przeciwieństwie do klasyfikatora, który przewiduje etykietę dla pojedynczej próbki bez uwzględniania sąsiednich próbek, CRF może uwzględnić kontekst. Aby to zrobić, przewidywania są modelowane jako model grafowy, który reprezentuje obecność zależności między przewidywaniami.

3.1.2 MLP + CRF

Dodanie warstwy MLP do CRF umożliwia modelowi lepsze wykorzystanie cech nieliniowych i złożonych wzorców w danych. MLP, będąc głęboką siecią neuronową, potrafi przetwarzać i rozpoznawać subtelne niuanse w danych, które mogą być pominięte w tradycyjnym modelu CRF. Wykorzystanie MLP w połączeniu z CRF może zatem zwiększyć dokładność prognoz poprzez lepsze modelowanie złożonych zależności.

3.1.3 LSTM + CRF

LSTM to rodzaj rekurencyjnej sieci neuronowej specjalizującej się w przetwarzaniu i przewidywaniu sekwencji danych z długoterminowymi zależnościami. Włączenie LSTM do CRF pozwala na skuteczniejsze zarządzanie informacjami o długoterminowych zależnościach w danych meteorologicznych. Jest to szczególnie przydatne w prognozowaniu pogody, gdzie warunki z przeszłości mogą mieć wpływ na przyszłe zjawiska.

3.2 ARIMA

Model ARIMA (Autoregressive Integrated Moving Average) jest modelem stosowanym do analizy szeregów czasowych. Może być użyteczny do predykcji przyszłych wartości danych zależnych od czasu. Część auto regresywna (AR) odpowiada za uwzględnienie zależności między obecnymi a poprzednimi wartościami szeregu. Część różnicowa (I) odpowiada za proces różnicowania, czyli pozbycie się sezonowości i trendu z danych. Część średniej ruchomej (MA) wygładza natomiast zaszumienie danych. Ze względu na te własności uznaliśmy, że przetestujemy jakość działania modelu ARIMA dla wybranych danych pogodowych.

3.3 BayesianLSTM

BayesianLSTM to rodzaj sieci LSTM, która wykorzystuje metody Bayesa do ilościowego określenia niepewności w przewidywaniach modelu. Tradycyjne sieci LSTM są deterministyczne – zapewniają pojedynczą prognozę, dla każdego wejścia. BayesianLSTM natomiast zapewniają prognozy probabilistyczne poprzez szacowanie rozkładu możliwych wyników oferując wgląd w niepewność tych prognoz.

3.3.1 Model

Model składa się z dwóch warstw LSTM oraz warstwy w pełni połączonej, dodatkowo po każdej warstwie LSRM zastosowano dropout.

3.3.2 Monte Carlo Dropout

Metoda stosowana do przybliżonego wnioskowania bayesowskiego w sieciach neuronowych. Polega ona na stosowaniu dropoutu zarówno w fazie uczenia, jak i wnioskowania. Poprzez utrzymanie włączonego dropoutu w fazie wnioskowania i uruchamiając model wielokrotnie, można uzyskać rozkład predykcji. Rozkład ten, kolejno można wykorzystać do oszacowania niepewności predykcji.

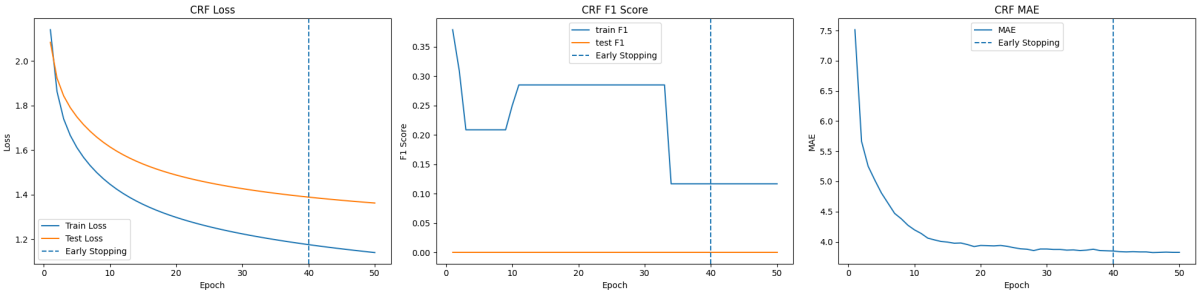
4 Eksperymenty

4.1 CRF

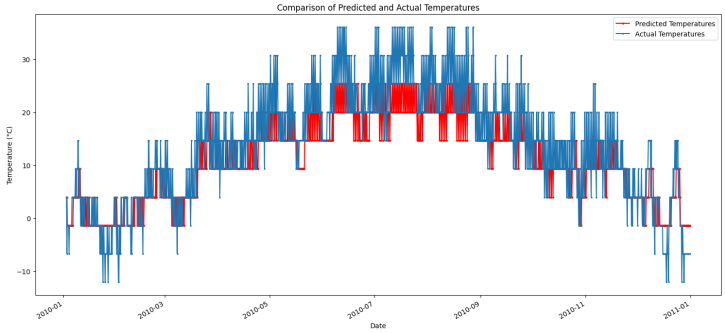
Wykorzystaliśmy implementację modelu CRF z zajęć, która była napisana do zadania klasyfikacji. Aby wykorzystać ten model do predykcji pogody podzieliliśmy występujące temperatury na zakresy. Testy przeprowadziliśmy dla 10 zakresów (czyli dla 10 klas).
Dane użyte do modelowania pochodziły z pomiarów temperatury z ostatniej doby, przy czym jako cechy wejściowe do modelu wybraliśmy temperatury z czterech wcześniejszych pomiarów. To stanowi nasze okno czasowe, które umożliwia modelowi uwzględnienie kontekstu temporalnego przy przewidywaniu przyszłych wartości.
Model trenowany był na zbiorze danych, który został podzielony na zbiór treningowy i testowy w proporcji 80/20. Jako, że mamy do czynienia z danymi sekwencyjnymi podzieliliśmy zbiór danych chronologicznie.

4.1.1 CRF

Model CRF z zajęć laboratoryjnych nieco dostosowaliśmy do zadania predykcji pogody. Funkcję celu zdefiniowaliśmy jako negative log-likeligood (NLL-Loss). Wyniki zostały zwizualizowane poprzez krzywe uczenia, które pokazują zmiany w F1 score i stracie na zbiorze treningowym i testowym oraz średni błąd bezwzględny na testowym.



Rysunek 1: Przebieg trenowania modelu



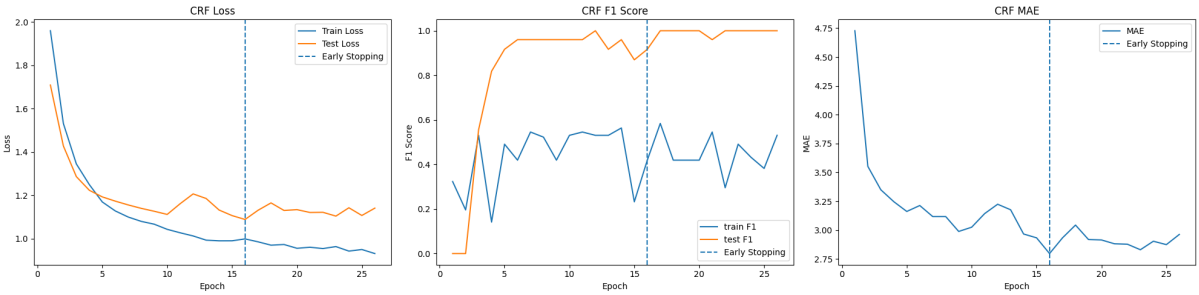
Rysunek 2: Predykcje modelu

Nasz model CRF po nauczeniu się uzyskał wyniki Test:

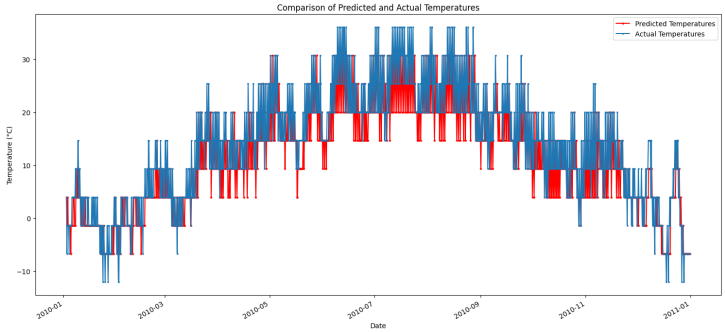
- NLL-Loss = 1.36,
- Accuracy = 0.38,
- F1 Score = 0.17
- Średni błąd bezwzględny (MAE) = 3.82°C

4.1.2 MLP + CRF

W tym modelu, CRF został połączony z MLP, aby zwiększyć zdolność modelu do uczenia się nieliniowych zależności między cechami. MLP składa się z kilku warstw perceptronów, gdzie każda kolejna warstwa uczy się coraz bardziej złożonych reprezentacji danych wejściowych. Warstwa wyjściowa MLP dostarcza emisje do CRF, który dokonuje końcowej klasyfikacji sekwencji.



Rysunek 3: Przebieg trenowania modelu



Rysunek 4: Predykcje modelu

Przeprowadziliśmy również badanie wpływu hiperparametrów na działanie modelu. Testowaliśmy:

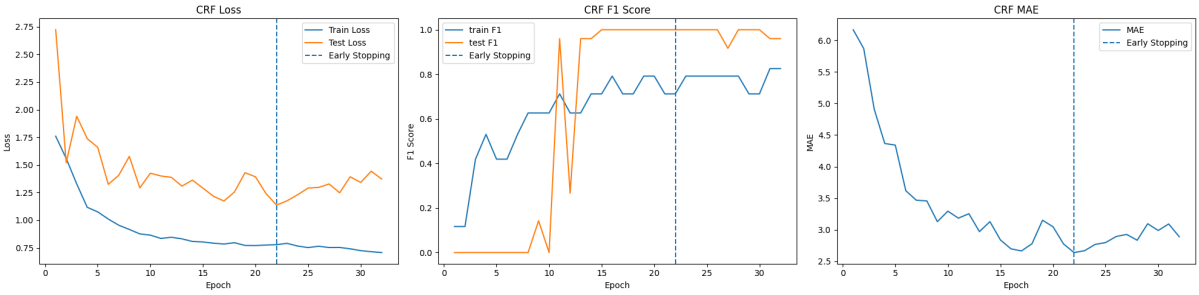
- Rozmiary warstw ukrytych:
  - Jednowarstwowy MLP: $[hidden\_dim, 32]$, $[hidden\_dim, 64]$, $[hidden\_dim, 128]$,
  - Dwuwarstwowy MLP: $[hidden\_dim, 32, 32]$, $[hidden\_dim, 64, 64]$, $[hidden\_dim, 128, 128]$,
- Współczynnik uczenia: $0.01, 0.001, 0.0001$,
- Rozmiar batcha: $16, 32, 64$.

Finalnie najlepsze wyniki pod względem błędu dał nam model o parametrach: $hidden\_sizes = [hidden\_dim, 64, 64], lr = 0.001, batch\_size = 32$.
Wyniki:

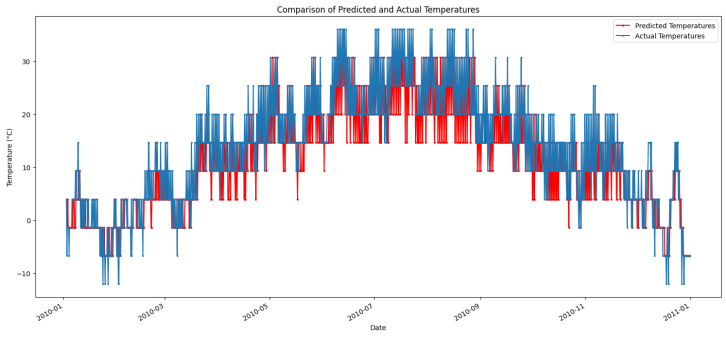
- NLL-Loss = 1.08,
- Accuracy = 0.51,
- F1 Score = 0.63
- Średni błąd bezwzględny (MAE) = 2.8°C

4.1.3 LSTM + CRF

CRF z LSTM łączy zalety CRF w modelowaniu zależności sekwencjalnych z umiejętnością LSTM do przetwarzania długotrwałych zależności w danych.



Rysunek 5: Przebieg trenowania modelu



Rysunek 6: Predykcje modelu

Przeprowadziliśmy również badanie wpływu hiperparametrów na działanie modelu. Testowaliśmy:

- Wymiary ukryte: $32, 64, 128$,
- Liczba warstw: $1, 2, 3$,

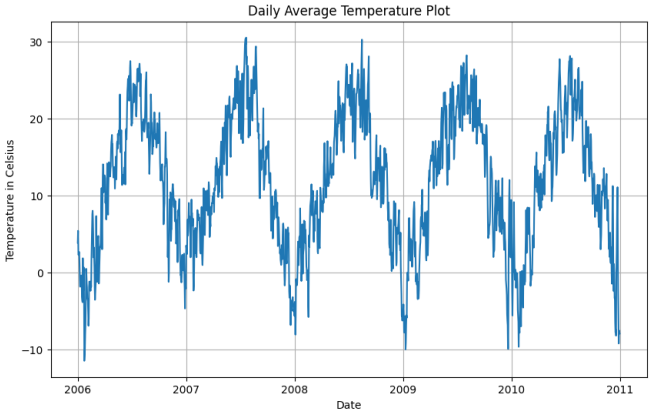
- Współczynnik uczenia: $0.01, 0.001, 0.0001$,
- Rozmiar batcha: $16, 32, 64$.

Finalnie najlepsze wyniki pod względem błędu dał nam model o parametrach: $hidden\_dim = 128, num\_layers = 3, lr = 0.01, batch\_size = 64$.
Wyniki:

- NLL-Loss = 1.10,
- Accuracy = 0.56,
- F1 Score = 0.86
- Średni błąd bezwzględny (MAE) = 2.44°C

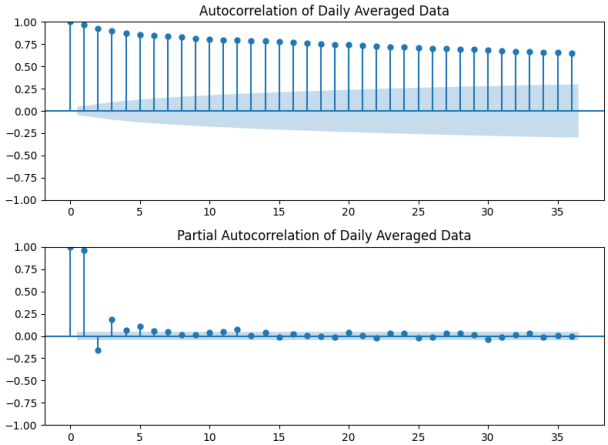
4.2 ARIMA

Wpierw dokonaliśmy przetwarzenia wstępnego i wczytaliśmy dane. Posługiwaliśmy się uśrednionymi danymi dziennymi.



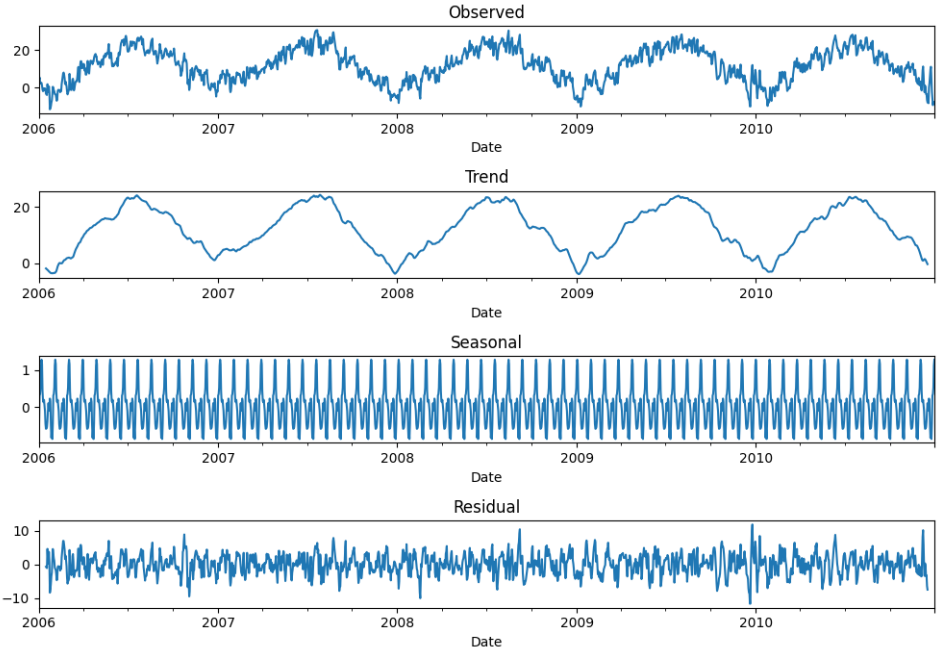
Rysunek 7: Uśredniona dzienna temperatura

Zbadaliśmy autokorelację i częściową autokorelację dziennych danych aby ocenić ich zależność w celu zbadania stacjonarności.



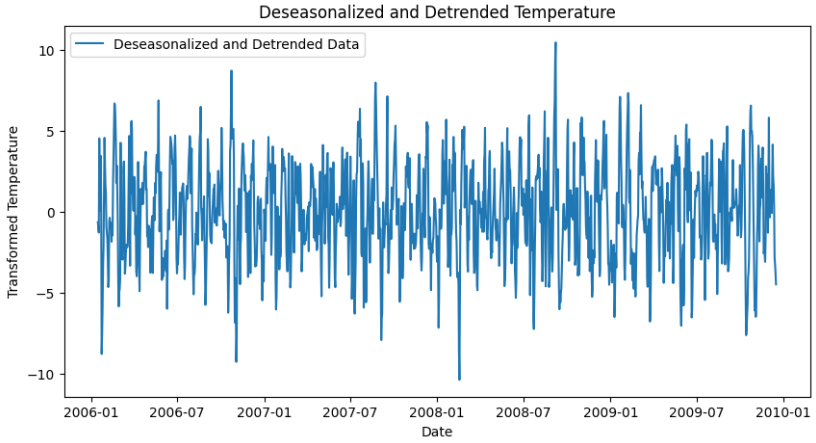
Rysunek 8: Autokorelacje danych

Z wykresu 8 obu korelacji wynika, iż dane nie są skorelowane a sam szereg nie jest stacjonarny. Dodatkowo przeprowadzając test Dickey'a-Fullera otrzymaliśmy p-wartość równą 0.078862. Dla p-wartości mniejszych od 0.05 szereg byłby stacjonarny. Zwizualizowaliśmy również trend oraz sezonowość szeregu.



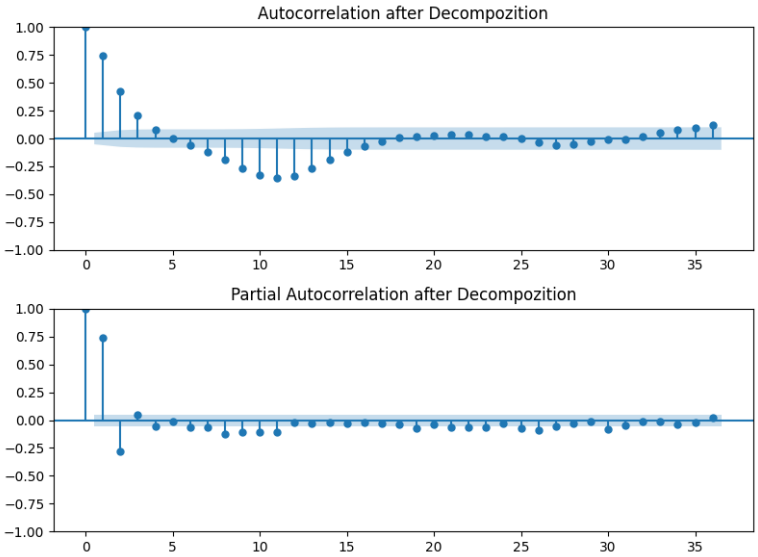
Rysunek 9: Dekompozycja szeregu

W celu pozbycia się wyraźnej sezonowości przeprowadziliśmy dekompozycję Walda dla danych aby uzyskać szereg stacjonarny. Usunęliśmy zarówno trend oraz sezonowość za pomocą różnicowania.



Rysunek 10: Dane po dekompozycji

Przetestowaliśmy stacjonarność zdekomponowanego szeregu.

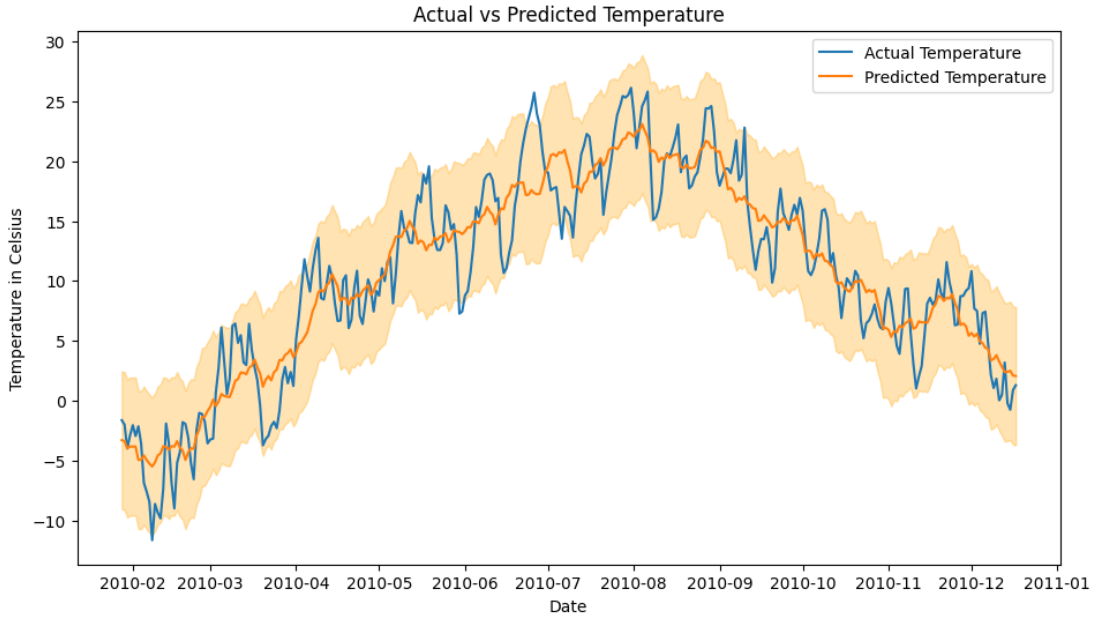


Rysunek 11: Autokorelacja danych po dekompozycji

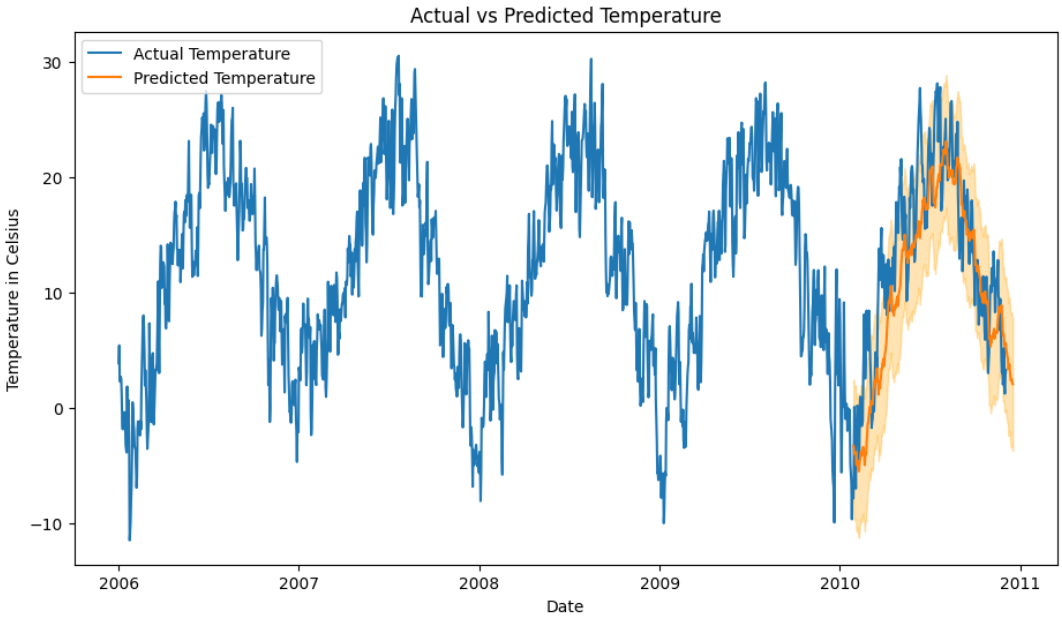
Zauważyliśmy poprawę autokorelacji na podstawie wykresu 11 oraz po ponownym przeprowadzeniu testu Dickey'a–Fuller uzyskaliśmy wartość poziomu krytycznego na poziomie zera. Taki szereg mogliśmy uznać za stacjonarny. Następnie dokonaliśmy testowania dopasowania szeregu ARIMA dla różnych parametrów p i q. Wykorzystaliśmy parametry z zakresu $0:5$, a najlepszy model wybierany był na podstawie kryterium AIC. Najlepsze dopasowanie uzyskaliśmy dla szeregu ARIMA(3,2):

$$y_t = 0.0023 + 1.4135 \cdot y_{t-1} - 0.3356 \cdot y_{t-2} - 0.1531 \cdot y_{t-3} - 0.5300 \cdot \epsilon_{t-1} - 0.4688 \cdot \epsilon_{t-2} + \epsilon_t$$

Kolejnym krokiem było wykonanie predykcji na zbiorze testowym, z uwzględnieniem przedziałów ufności na poziomie ufności 0.95.



Rysunek 12: Predykcje temperatury na 2010 rok

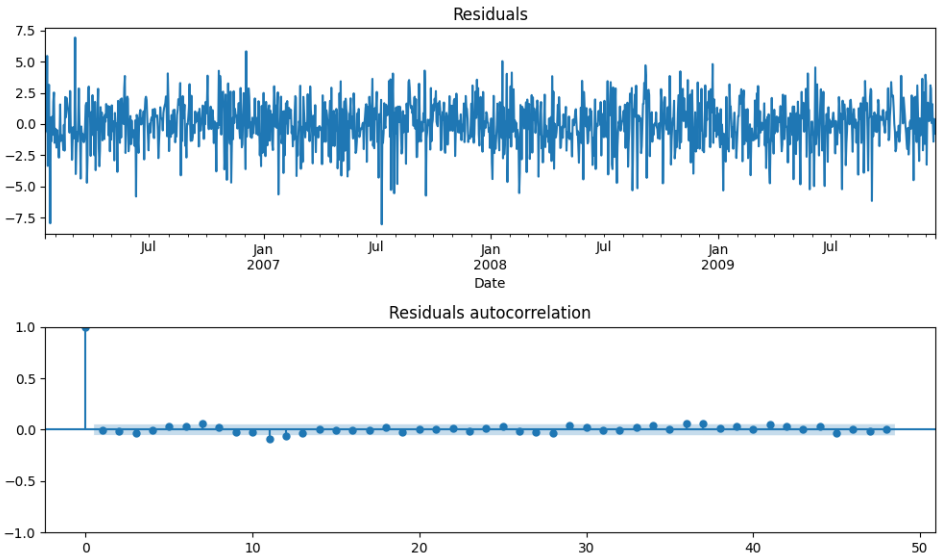


Rysunek 13: Predykcje temperatury

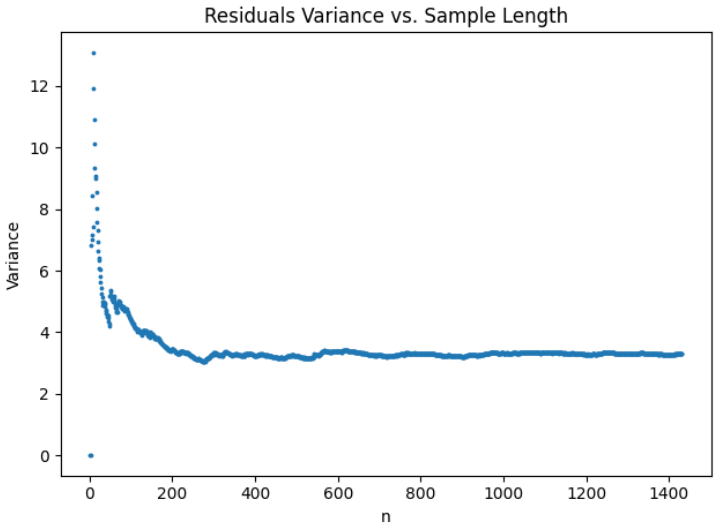
Miary jakości predykcji:

- Średni błąd bezwzględny (MAE) = 3.48
- Błąd średniokwadratowy (MSE) = 18.13

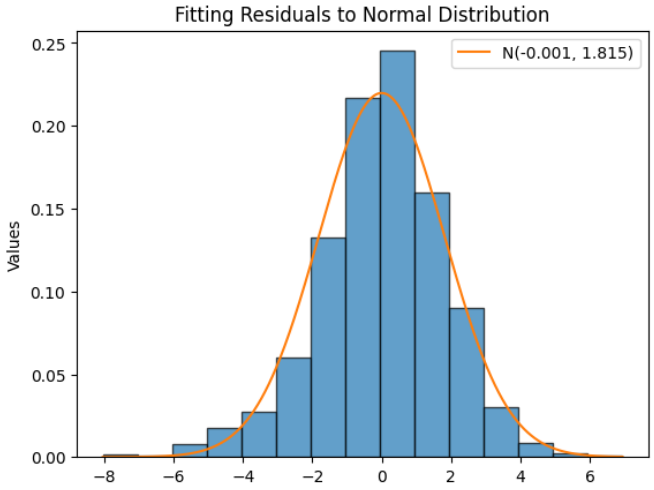
Aby ocenić jakość dopasowania modelu przeprowadziliśmy również analizę residuów i zweryfikujemy założenia dotyczące szumu.



Rysunek 14: Residua oraz ich autokorelacja



Rysunek 15: Wariancja residuów



Rysunek 16: Dopasowanie residuów do rozkładu normalnego

Residua spełniły założenia odnośnie średniej bliskiej zera, stałości wariancji w zależności od próby oraz dopasowały się dobrze do rozkładu normalnego.

4.3 BayesianLSTM

W celu zastosowania architektury BayesianLSTM, należało w pierwszym kroku odpowiednio sformatować dane. Zmienne kategoryczne (tj.: *summary* i *preciptype*) zakodowano za pomocą *OneHotEncoding*, natomiast zmienne numeryczne (tj.: temperatura, odczuwalna temperatura, wilgotność, prędkość wiatru, nośność wiatru, widoczność, ciśnienie, godzinę i miesiąc) zostały przeskalowanie za pomocą $MinMaxScaler$ do wartości $X \in [0, 1]$. Finalnie zbiór danych liczył 26 cech. Następnie, podobnie jak powyżej, stworzono zbiór treningowy i testowy w proporcjach 80/20, wykorzystując do tego $SlidingWindow$ o dobowej szerokości okna (w przeciwieństwie do poprzedniego podejścia zawiera on wszystkie powyższe cechy).
Przeprowadzony został tuning hiperparametrów, w celu wybrania najlepszej konfiguracji sieci. W tym celu zastosowano przeszukiwanie siatki. Parametry, które były optymalizowane to: $hiden\_dim1$, $hidden\_dim2$, $stacked\_layers$, $learning\_rate$ oraz $batch\_size$. Każdy z nich miał 3 wartości, które zostały sprawdzone co przełożyło się na 243 modyfikacji.

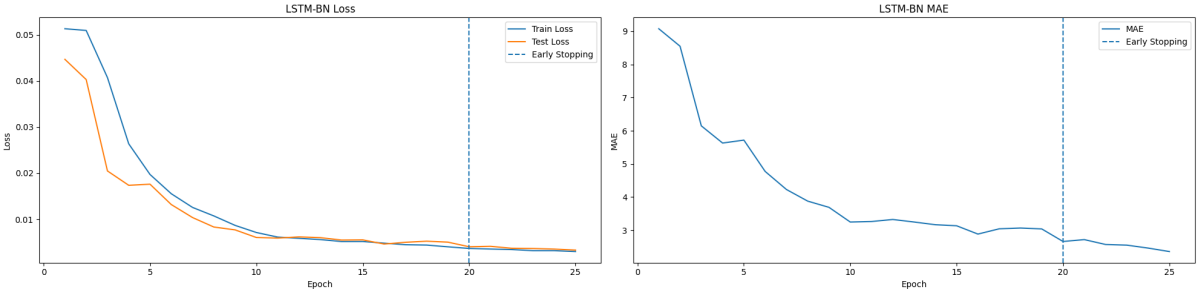
Testowano:

- Wymiary ukryte ($LSTM_1$): $64, 128, 256$,
- Wymiary ukryte ($LSTM_2$): $16, 32, 64$,
- Liczba warstw: $1, 2, 3$,
- Współczynnik uczenia: $0.01, 0.001, 0.0001$,
- Rozmiar batcha: $16, 32, 64$.

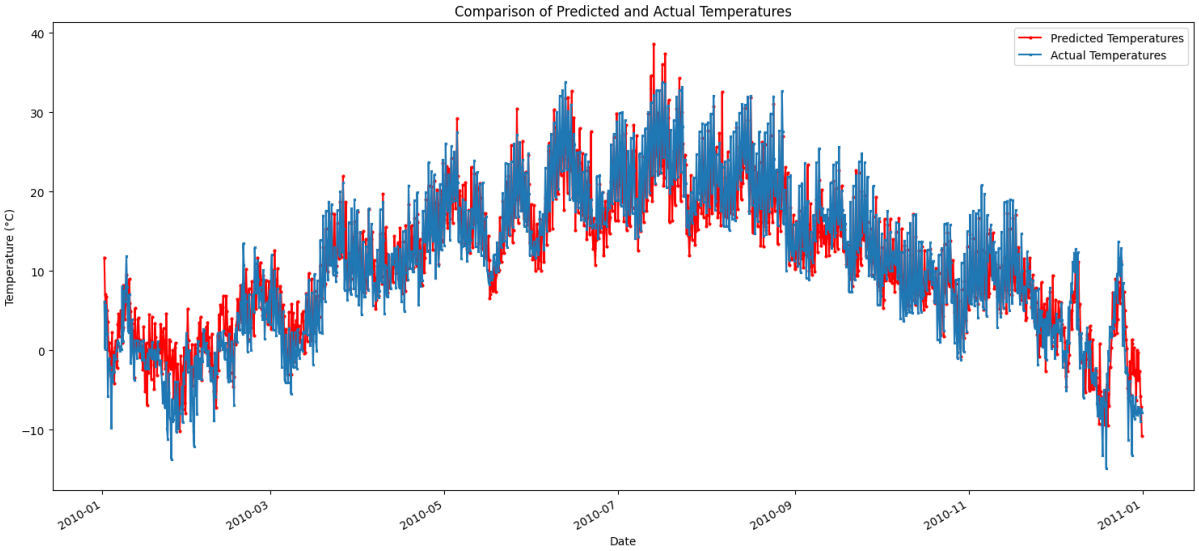
Finalnie najlepsze wyniki pod względem błędu dał nam model o parametrach: $hidden\_dim1 = 64, hidden\_dim2 = 32, num\_layers = 3, lr = 0.001, batch\_size = 32$.

Wyniki:

- MSE $= 9.02$,
- Średni błąd bezwzględny (MAE) $= 3.74$°C,

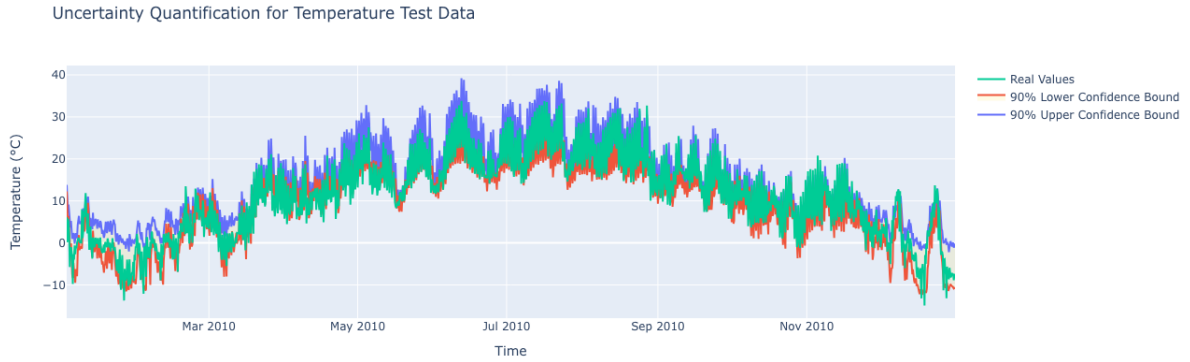


Rysunek 17: Przebieg trenowania modelu



Rysunek 18: Predykcja temperatury

Finalnie po wyborze najlepszego modelu, została zbadana, wyżej wspomniana, niepewność dla przedziału ufności 90% oraz naniesiona na predykcję.



Rysunek 19: Naniesiona niepewność modelu na wyniki predykcji

5 Porównanie wyników modeli

Porównując wyniki zastosowanych modeli do predykcji pogody, można zauważyć różnice w efektywności poszczególnych metod w odniesieniu do mierzonych metryk.

Metryka	CRF	CRF+MLP	CRF + LSTM	BayesianLSTM	ARIMA
Accuracy	0.38	0.51	0.56	—	—
F1 score	0.17	0.63	0.86	—	—
NLL Loss	1.36	1.08	1.10	—	—
MSE	27.07	19.15	17.47	9.02	18.13
MAE	3.8	2.8	2.44	3.67	3.48

W swojej pracy porównaliśmy różne modele do predykcji temperatury, każdy z nich miał różne podejścia i techniki, co przełożyło się na ich skuteczność. Oto porównanie ich wyników:

- Model CRF okazał się najmniej efektywny, co widoczne jest w najniższych wynikach metryk Accuracy oraz F1 score. Mimo to, oferuje on wgląd w modele sekwencyjne.
- Dodanie warstwy MLP do CRF znacząco poprawiło wyniki, zwiększając dokładność (Accuracy) i skuteczność (F1 score). Skuteczność ta jest potwierdzona również mniejszym błędem MSE.
- Integracja LSTM z CRF dalej poprawia wyniki, dostarczając najlepszy F1 score spośród modeli CRF. Radzi sobie również najlepiej względem MAE.
- BayesianLSTM osiąga najniższy błąd MSE natomiast prawie najwyższy MAE, sugeruje to, że model popełnia mniej 'ekstremalnych' błędów, ale więcej takich o umiarkowanej wielkości.
- ARIMA, mimo swojej tradycyjności w analizie szeregów czasowych, nie wykazał wyraźnej przewagi nad nowocześniejszymi modelami głębokiego uczenia, co widać w podobnym poziomie błędu MSE do CRF z LSTM.