Предсказване на валежи с размита невронна мрежа

Курсова работа по „Размити множества и приложения“,

От Тихомир Кожухарски, ф.н. 25387

# Въведение

Предсказването на времето е доста труден проблем, който изисква използването на сложни физични модели и обработката на големи количества данни, за да бъде решен със задоволителна точност, дори и с цел прогнозиране в краткосрочен план. Съвременните методи и хардуер позволяват правенето на 5-дневни прогнози, с 90% точност, на 7-дневни с 80% и на 10-дневни, с около 50% точност. В тази курсова работа ще разгледаме значително по-опростена задача, а именно – прогнозирането на това дали ще вали на следващия ден, на базата на метеорологични данни от днес.

# Използван алгоритъм

Ще приложим Adaptive Network-based Fuzzy Inference System (ANFIS) – невронна мрежа, с помощта на която могат да се научат размити множества и размити променливи от данни, без да е необходимо ръчно да се въвеждат такива. Имплементацията е направена с помощта на фреймуъркът за невронни мрежи PyTorch . Мрежата се състои от 5 слоя:

Diagram, schematic

Description automatically generated

* **Слой 1 - Fuzzification** – състои се от неврони, които представят размитите множества. Приемат за вход стойността на променливата, а изходът им е стойността на функцията на принадлежност.
* **Слой 2 – Premise** – броят на невроните е равен на броят на научените правила. Невронната мрежа, представена в тази курсова работа, съдържа всички възможни правила. Този слой не съдържа параметри. Използва умножение на входните данни, като Т-норма. Резултатът от него е силата, с която се активира съответното правило.
* **Слой 3 – Normalization** – Нормализира степента на активация на правилата, така че сумата от всички да бъде равна на 1
* **Слой 4 – Consequent** – Този слой имплементира изходът от Такаги-Сугено правилата, като линейни функции на входните данни, претеглени със степента на активация на съответното правило (изходът от слой 3)
* **Слой 5 – Summing** - Един неврон, използващ сумирането, като Co-норма. В нашият случай, сме използвали сигмоидна функция, като активация, защото задачата, която решаваме е от класификационен тип.

Имплементирани са следните функции на принадлежност: Singleton, Триъгълна, Трапецоидна и Гаусова.

# Данни

Използван е дейтасета “Rain in Australia”, който съдържа метеорологични дани за около 10 години – от 12.2008г, до 06.2017г., за различни градове в Австралия, като съдържа общо 145460 записа. Всеки запис има следните стойности:

* **Date** – дата на измерване
* **Location** – градът, в който са измерени данните
* **MinTemp** – минимална температура в съответния ден
* **MaxTemp** – максимална температура в съответния ден
* **Rainfall** – количеството валежи измерено в съответния ден, измерено в милиметри
* **Evaporation** – количеството на изпарения
* **Sunshine** – броят светли часове в този ден
* **WindGustDir** – посоката на най-силния порив на вятъра
* **WindGustSpeed** – силата на най-силния порив на вятъра
* **WindDir9am** – посоката на вятъра, в 09:00ч
* **WindDir3pm** – посоката на вятъра в 15:00ч
* **WindSpeed9am** – силата на вятъра в 09:00ч
* **WindSpeed3pm** – силата на вятъра в 15:00ч
* **Humidity9am** – влажността на въздуха в 09:00ч
* **Humidity3pm** – влажността на въздуха в 15:00ч
* **Pressure9am** – атмосферното налягане в 09:00ч
* **Pressure3pm** – атмосферното налягане в 15:00ч
* **Cloud9am** – каква част от небето е покрита с облаци в 09:00ч. Измерва се в осмини.
* **Cloud3pm** – каква част от небето е покрита с облаци в 15:00ч. Измерва се в осмини.
* **Temp9am** – температура на въздуха в 09:00ч
* **Temp3pm** – температура на въздуха в 15:00ч
* **RainToday** – дали е валяло този ден
* **RainTomorrow** – дали е валяло на следващия ден

# Обработка

Стъпките по изчистването на данните са подробно описани в следния jupyter notebook: <https://github.com/tihomir-kojuharski/anfis/blob/master/data_analysis.ipynb>

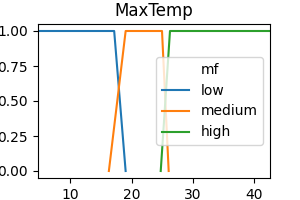
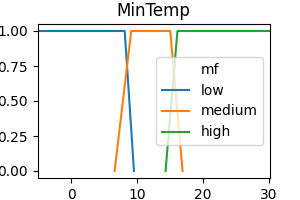
Накратко, направна е следната обработка на данните:

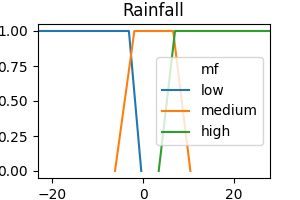
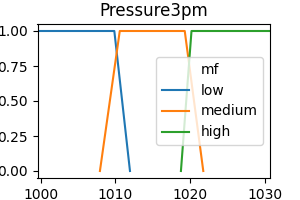
* Колоните, съдържащи посока на вятъра (WindGustDir, WindDir9am, WindDir3pm) са преобразувани от номенклатурни (S, N, NE, ..) в числени – от 0 до 360 градуса
* Добавени са две нови колони Latitude и Longitude, които са извлечени на базата на Location, с идеята да могат да се научат размити множества и правила, свързани с локацията
* Попълнени са част от липсващите стойности в колоните MinTemp, MaxTemp, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Temp9am, Temp3pm, като за целта данните бяха групирани по локация, след което беше направена интерполация на съответните стойности, въз основа на данните от дните, за които имаме стойности
* Попълнени са част от липсващите липсващи стойности в колоните WindGustDir, WindDir9am и WindDir3pm, отново с интерполация, но само в случаите, в които наличните стойности за съседните дни са били близки
* Колоните Evaporation, Sunshine, Cloud9am и Cloud3pm бяха премахнати, тъй имаха прекалено много липсващи стойности
* Премахнати са всички записи, които след описаната обработка са имали поне една липсваща стойност за някоя колона

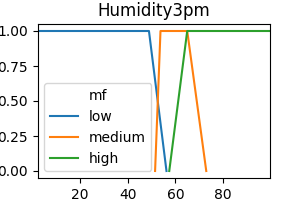
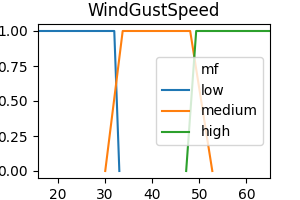
# Експерименти

Бяха направени множество експерименти, с използването на различни характеристики от данните. Експериментите бяха ограничени от факта, че броят научени размити правила нараства експоненциално спрямо броят характеристики, поради избраната архитектура на невронната мрежа.

Най-добрият резултат се получи с използването на следните променливи, като за всички цифрови стойности бяха избрани по три отворени трапецоидни размити множества, които да представят съответно „малка“, „средна“ и „голяма“ стойност:





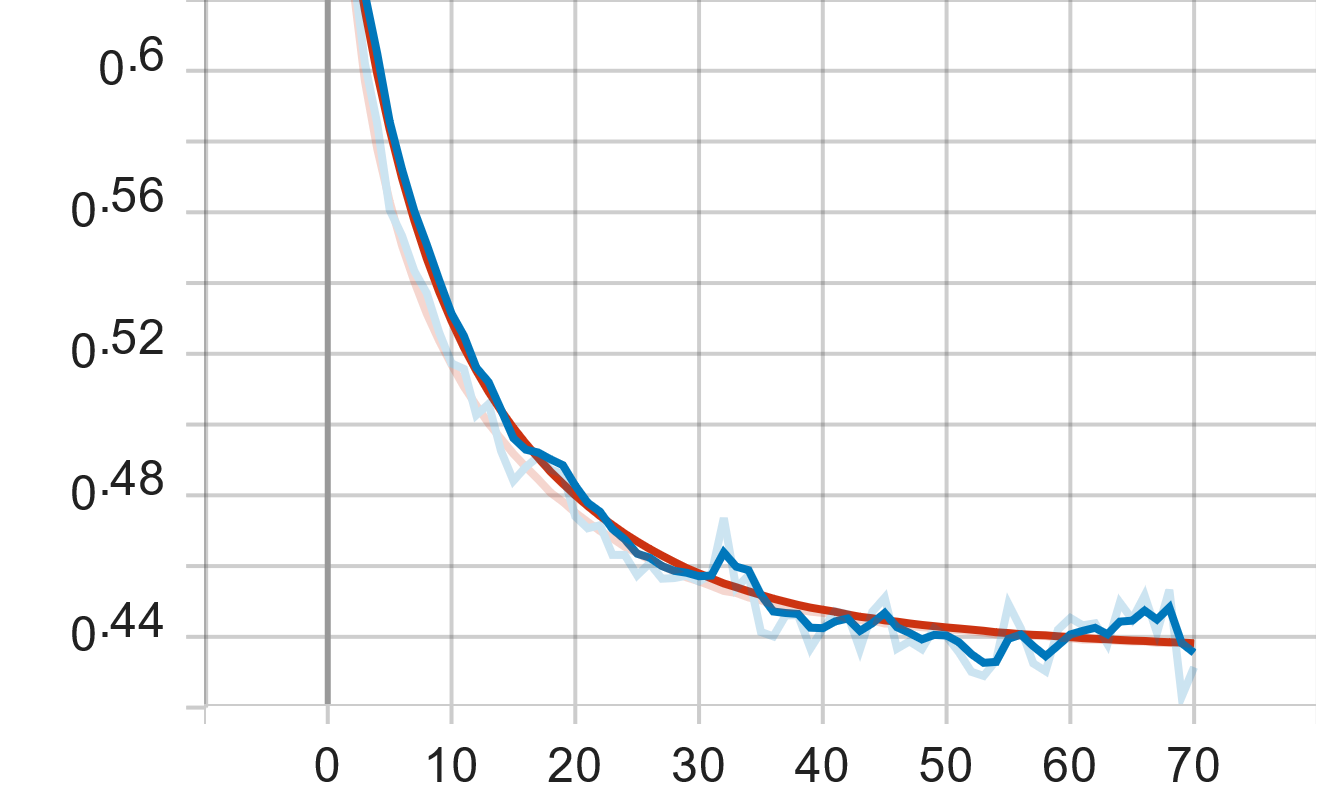


* Както и 2 singleton функции за RainToday – True и False

Бяха научени 1458 Такаги-Сугено размити правила – всички възможни правила, които разглеждат всички комбинации от научените размити множества.

Моделът беше трениран за 70 епохи, с learning rate 0.01 и batch-size 512, използвайки Stochastic Gradient Descent с momentum 0.9. За loss функция беше използвана Binary Cross Entropy Loss функцията.

Графика на loss функцията по епохи, за train dataset-а (в син цвят) и validation dataset-a (в оранжев цвят).



## Получените резултати:

**Precision:** 0.6418994665145874

**Recall:** 0.26083996891975403

**F1:** 0.3709443509578705

# Изводи

Използваният модел е доста опростен, и не всзема предвид всички свойства, налични в данните. Моделът разглежда всички възможни правила, включващи всички комбинации от променливи, поради което броят на правилата нараства експоненциално спрямо броя промеливи.

# Идеи за бъдещи подобрения:

* Ограничаване на правилата, само до определени комбинации от променливи, така че да могат да се използват повече от свойствата от базата данни
* Експериментиране с нелинейни изходящи функции в Такаги-Сугено правилата

# Използвана литература:

* ANFIS : Adaptive-Network-Based Fuzzy Inference System - <https://liacs.leidenuniv.nl/~nijssensgr/CI/2011/anfis.pdf>