HACKATON 2021

Karamelizovana Karakondžula

Mihailo Jovanović, Božidar Mitrović
Marija Erić, Nikola Andrić Mitrović

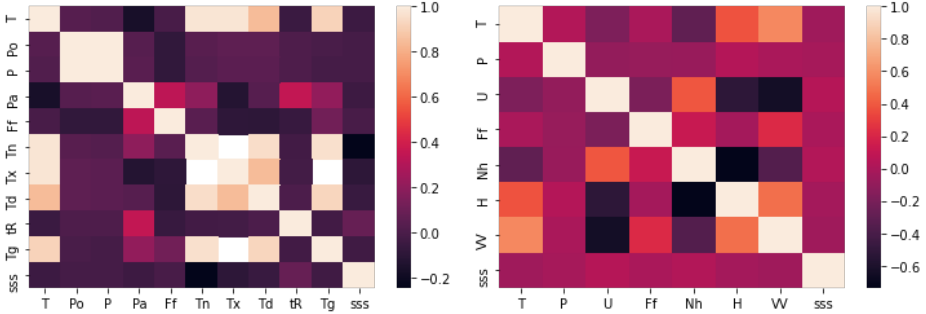
Matematički fakultet

April 4, 2021

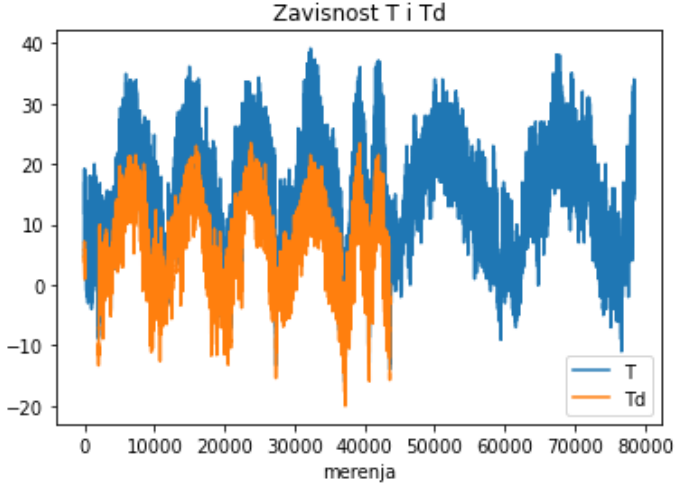
Sadržaj

1. Preprocesiranje
   - Belgrade Airport Dataset
   - Zagadjenost vazduha -NBG i SG
2. Model
3. Pollute or Salute

Belgrade Airport Datase - Koorelacija izmedju numeričkih vrednosti



Koorelacija izmedju numeričkih vrednosti



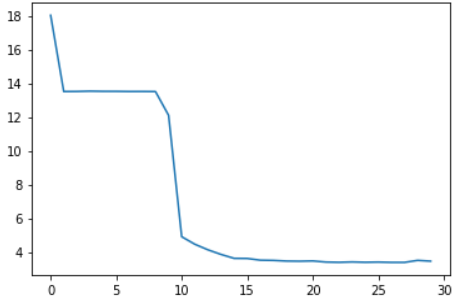
Belgrade Airport Dataset

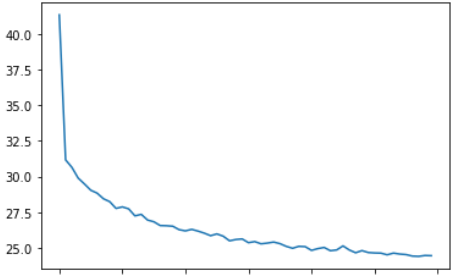
- Tg, Td, Tx, Tn jako korelisane sa T
- Premalo informacija(tG) ili nerelevantni(E, E.)
- DD i U pomešani podaci
- N u Nh transformisali u numerički iz mešovitog
- Ff u VV transformisali u numerički i predvidjali
- DD(smer vetra) u kategoricke

Predvidjanje vrednosti atributa Ff i VV

```
FillModel(
  (fnn): Sequential(
    (0): Linear(in_features=23, out_features=32, bias=True)
    (1): ReLU()
    (2): Dropout(p=0.2, inplace=False)
    (3): Linear(in_features=32, out_features=16, bias=True)
    (4): ReLU()
    (5): Dropout(p=0.2, inplace=False)
    (6): Linear(in_features=16, out_features=1, bias=True)
    (7): ReLU()
  )
)
```

Predvianje Ff i VV

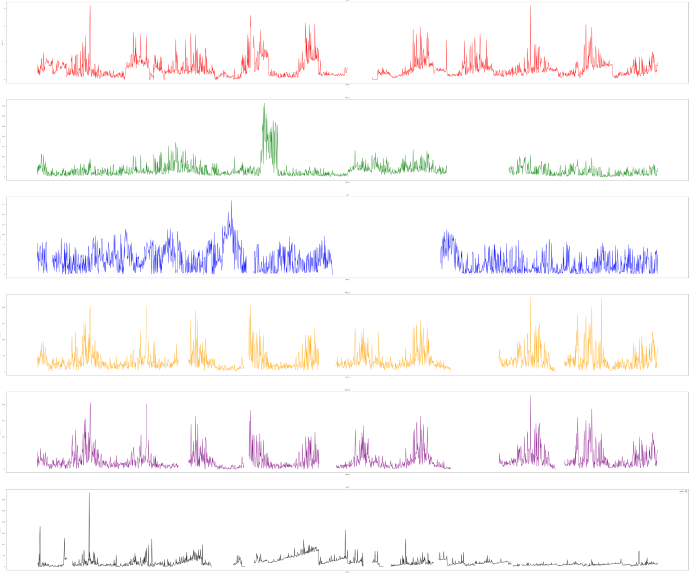




Zagadjenost vazduha -NBG i SG

- Sortiranje po datumu
- Uklanjanje autlajera(grešaka pri merenju)
- Spajanje tabela radi smanjenja broja nedostajućih vrednosti
- Linearna interpolacija

Vrednosti nakon uklanjanja autlajera



Spajanje tabela

```
data2.isna().sum()
```

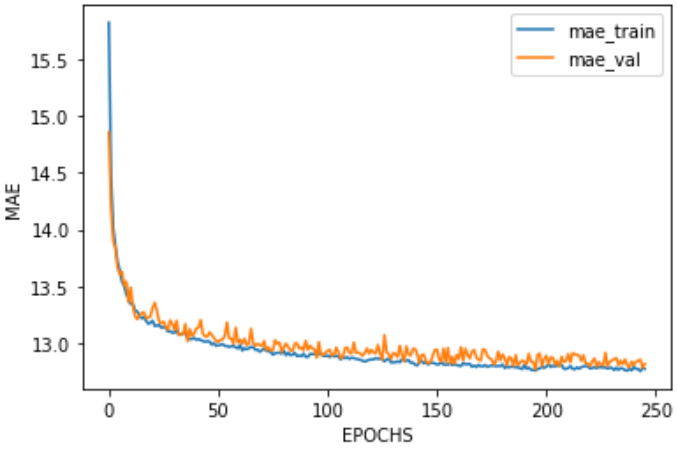
```
Datum_i_Vreme         0
MernaStanica          0
B                 51179
CO                 4766
NO2               10358
O3                19363
PM10              13955
PM25              13912
SO2                9820
dtype: int64
```

```
Datum_i_Vreme         0
CO                    9
NO2                6164
O3                 3375
PM10               4623
PM25               4623
SO2                5269
dtype: int64
```

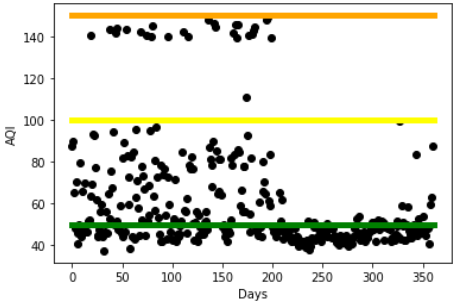
Arhitektura mreže

```
FullyConnected(
  (fnn): Sequential(
    (0): Linear(in_features=25, out_features=64, bias=True)
    (1): ReLU()
    (2): Dropout(p=0.2, inplace=False)
    (3): Linear(in_features=64, out_features=128, bias=True)
    (4): ReLU()
    (5): Dropout(p=0.2, inplace=False)
    (6): Linear(in_features=128, out_features=64, bias=True)
    (7): ReLU()
    (8): Dropout(p=0.2, inplace=False)
    (9): Linear(in_features=64, out_features=6, bias=True)
    (10): ReLU()
  )
)
```

Treniranje modela



Air Quality Index



$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - C_{low}) + I_{low}$$

I = the (Air Quality) index,

C = the pollutant concentration,

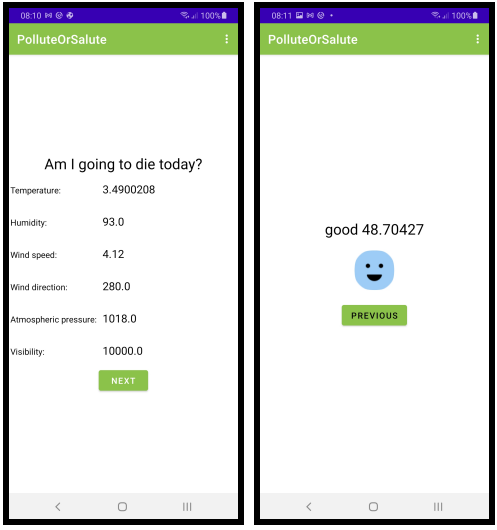
C_{low} = the concentration breakpoint that is $\leq C$,

C_{high} = the concentration breakpoint that is $\geq C$,

I_{low} = the index breakpoint corresponding to C_{low},

I_{high} = the index breakpoint corresponding to C_{high}.

Pollute or Salute



Pollute or Salute

```
public float[] predict(float temperature, float atmosphericPressure, float humidity, float windSpeed, float clouds, float visibility, float windDirection){

    Tensor tensor = preProcess(temperature, atmosphericPressure, humidity, windSpeed, clouds, visibility,  windDirection);

    IValue inputs = IValue.from(tensor);
    Tensor outputs = model.forward(inputs).toTensor();

    float[] output = outputs.getDataAsFloatArray();

    return output;
}
```

```
public Tensor preProcess(float temperature, float atmosphericPressure, float humidity, float windSpeed, float clouds, float visibility, float windDirection) {
    if(visibility > 10000)
        visibility = 10000;

    visibility /= 100;


    float[] angles = {90, 77.5f, 112.5f, 0, 45, 22.5f, 337.5f, 315, 180, 135, 157.5f, 202.5f, 225, 270, 292.5f, 247.5f};

    int index = 0;

    if(windDirection > 347.5){
        index = 3;
    }else {
        for (int i = 1; i < angles.length; i++) {
            if (abs(angles[i] - windDirection) < 11.25) {
                index = i;
            }
        }
    }

    float[] tempArr = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
    tempArr[index] = 1f;

    float[] data = {temperature,atmosphericPressure, humidity, windSpeed, clouds, visibility,0, 0,
            tempArr[0],tempArr[1],tempArr[2],tempArr[3],
            tempArr[4],tempArr[5],tempArr[6],tempArr[7],
            tempArr[8],tempArr[9],tempArr[10],tempArr[11],
            tempArr[12],tempArr[13],tempArr[14],tempArr[15], 0};
    long[] shape = {1,25};
    return Tensor.fromBlob(data, shape);
}
```

Hvala na pažnji!