Faculty of computer science

Master of data science

Moscow, 2022

ML-based plant type classification based on time series agricultural data

Student: Elizaveta Gavrilova
Supervisors: Polina Polunina, Visiting Lecturer, Big Data and Information Retrieval Department, Faculty of Computer Science; Specialist McKinsey and co,
Sarkis Samvelovich Grigoryan, Head of Competence center for Artificial Intelligence at Digital Economy Development Fund,

Problem
wheat/no-wheat classification for pre-defined NDVI

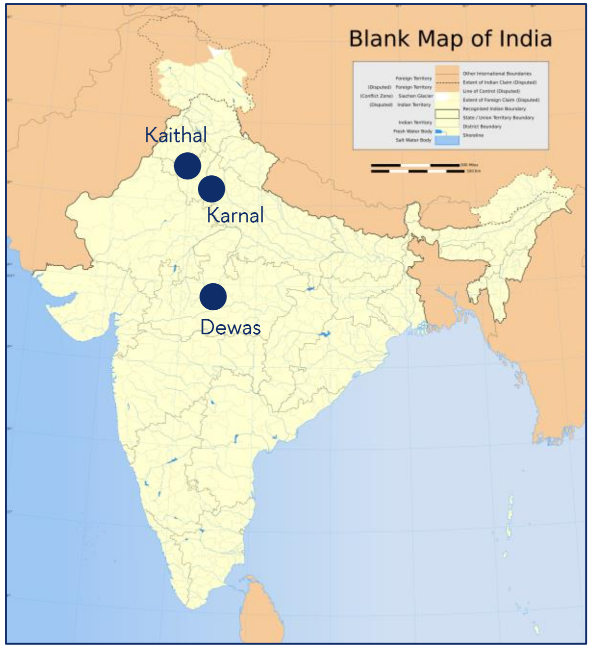
Data
- spectral data (NDVI)
- non-standard structure
- small sample due to NDA restrictions

Partner
GC «2050, DIGITAL»

Goal
find best model in terms of classification metrics

Data description

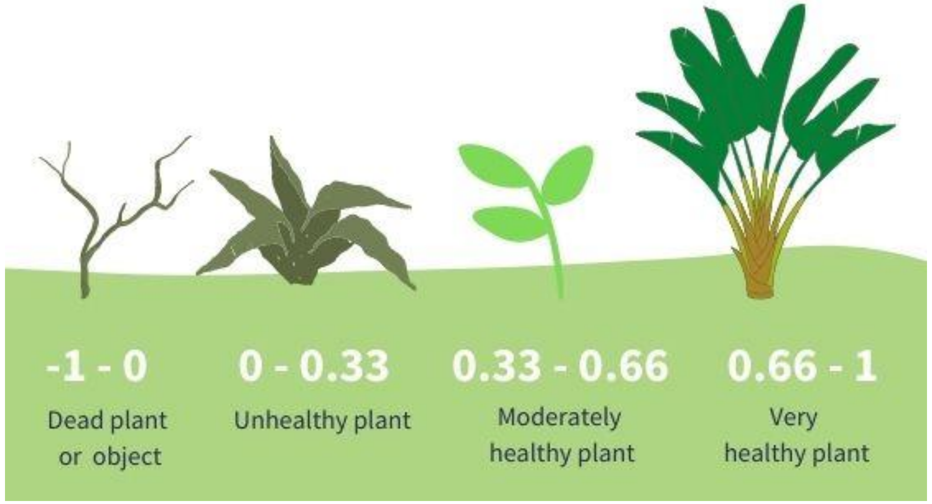
3 Indian districts: Kaithal, Karnal and Dewas



Source: OneSoil

~200 data points collected for each district
Since 2020-10-20 to 2021-05-10

Data points represent NDVI indices



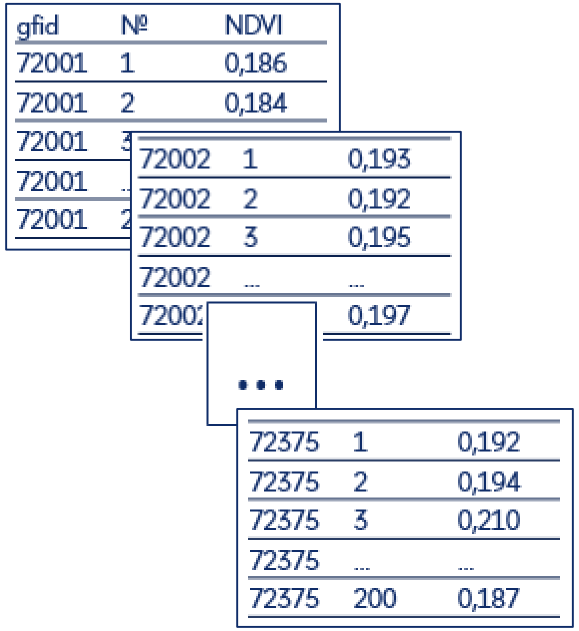
Source: Up42

Data description

gfid	№	NDVI
72001	1	0,186
72001	2	0,184
72001	3	0,183
72001	…	…
72001	200	0,185
72002	1	0,193
72002	2	0,192
72002	3	0,195
72002	…	…
72002	200	0,197

The dataset looks like the following: At each place(gfid) NDVI index was measured 200 times. Then, all those measurements were concatenated.

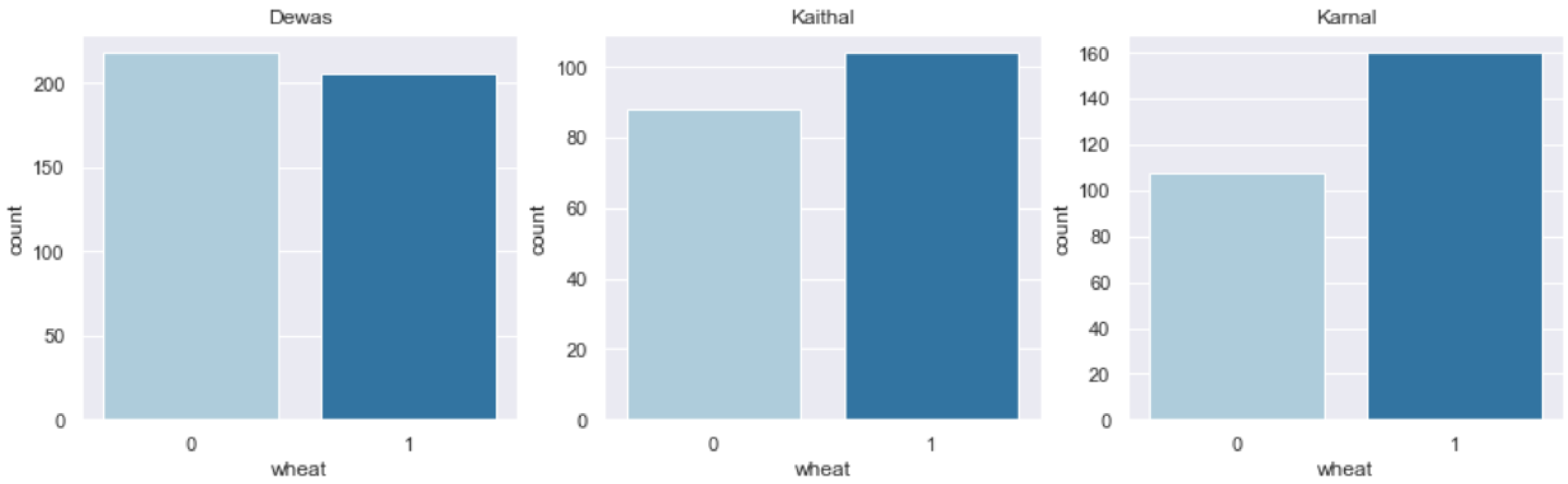
So, originally the data is a 3d matrix with dimensions:
- place(gfid)
- timestamp(1 … 200)
- NDVI



Objectives | Dataset | Methods | Experiments | Results | Outcomes

Data description

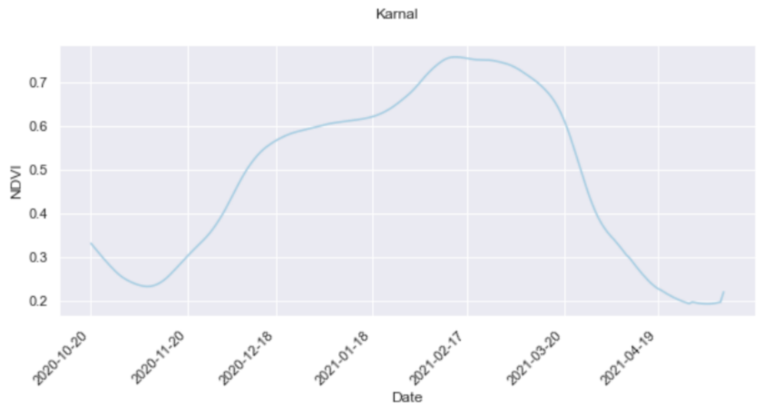
Target distribution: imbalance differs for the districts
Splitting strategy: random stratified 20/80

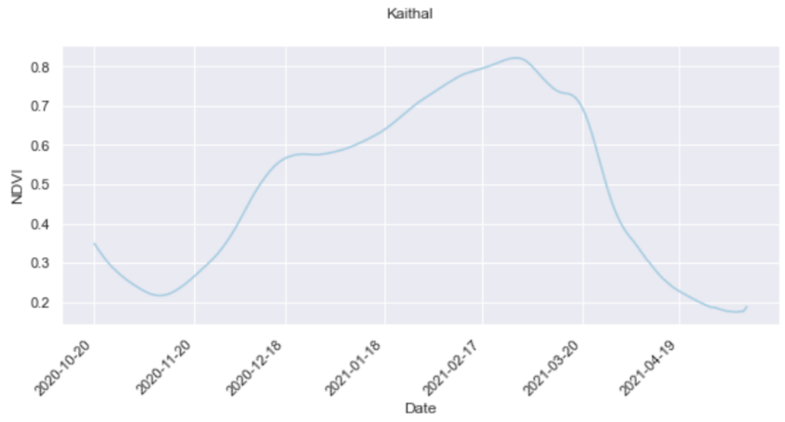


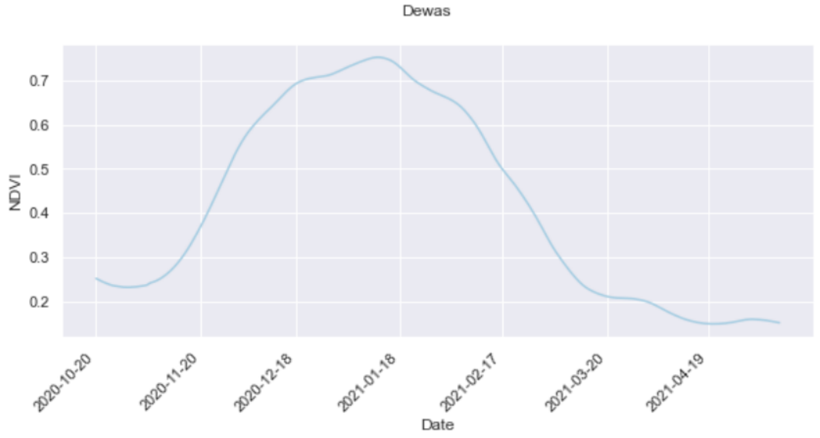
100 villages | 83 villages | 75 villages

Data description

Averaged time serieses for each district







ARIMA

Auto Regressive Integrated Moving Average (**ARIMA**) model is among one of the most popular and widely used statistical methods for time-series forecasting.
It is a class of statistical algorithms that captures the standard temporal dependencies that is unique to a time series data in order to predict future trends.

ARIMA models has 3 parameters: **q**, **d** and **p**,

- **q** represents **AR** component order,
- **d** stand for order of the integrated series,
- **p** represents **MA** component order,

Model equation is

$$\hat{y_t} = c + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q} + e_t$$

ARIMA model assumptions:

- time series is stationary,
- residuals are homoscedastic,
- residuals are normally distributed,

ARIMA

Assumptions testing

Stationarity
Dickey-Fuller test

H_0: there is unit root in time series and the series is non-stationary
H_a: no unit roots in time series and the series is stationary

Distribution: Dickey-Fuller's

Normality
D'Agostino-Pearson test

H_0: the sample is drawn from a normally distributed population
H_a: the sample is not drawn from a normally distributed population

Distribution: Chi-square

Homoscedasticity
Ljung–Box test

H_0: the data is independently distributed, the correlations in the population from which the sample is taken are 0
H_a: the data is not independently distributed, serial correlation exists,

Distribution: Chi-square

Visual tests: QQ, Auto-correlation and Partial auto-correlation plots

Classical ML methods

Logistics Regression

Linear algorithm,
In statistics, the logistic model is a statistical model that models the probability of one event (out of two alternatives) taking place by having the log-odds for the event be a linear combination of one or more independent variables (predictors),

Random Forest

Decision trees algorithm, Random forests use a method called bagging to combine many decision trees to create an ensemble, Bagging simply means combining in parallel,

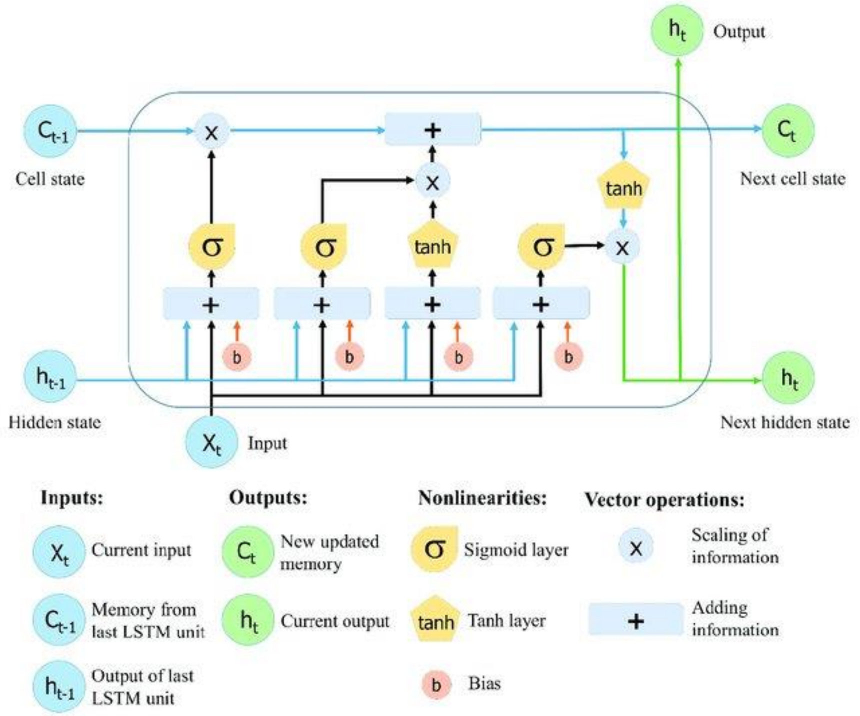
Boosting on trees

Decision trees algorithm, In boosting, new trees are formed by considering the errors of trees in previous rounds, Therefore, new trees are created one after another, Each tree is dependent on the previous tree,

SVM

SVM maps training examples to points in space so as to maximise the width of the gap between the two categories, New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall,

RNN: LSTM



Source: ResearchGate

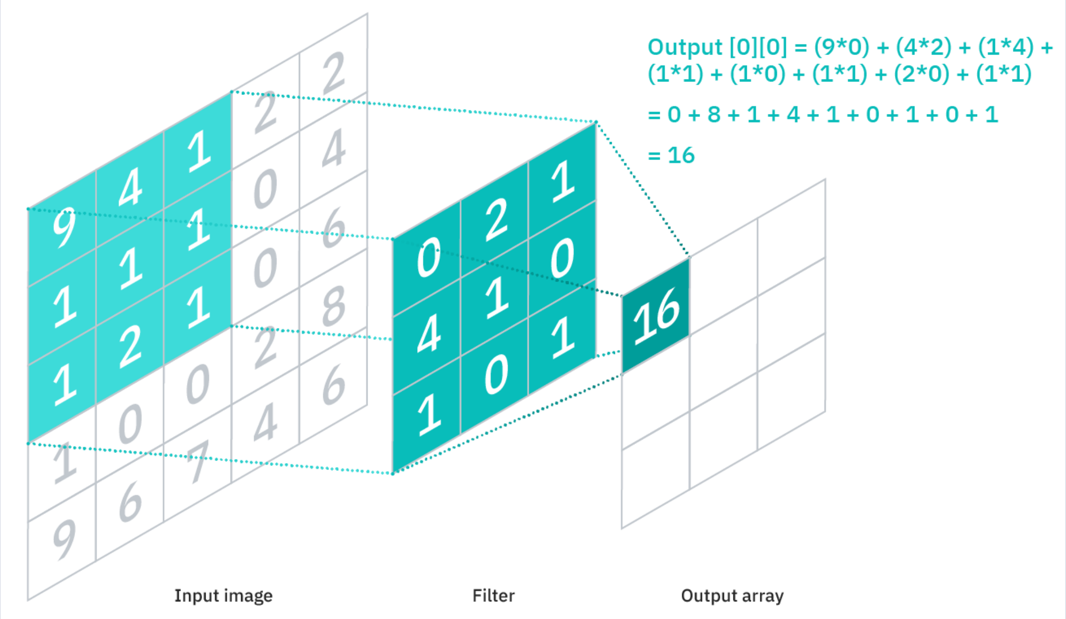
Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies,

The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called **gates**,

Gates are a way to optionally let information through, They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

CNN

Convolutional neural network is composed of multiple building blocks, such as **convolution layers**, **pooling layers**, and **fully connected layers**, and is designed to automatically and adaptively learn spatial hierarchies of features.



Source: IBM

ARIMA data preparation

Given: for each data point ~200 observations

For example, at some place in Dewas with id 72001, we have a time series of collected NDVI:

	NDVI
1	0,186
2	0,184
3	0,183
…	…
200	0,185

Based on visual PAC and AC analysis and Dickey-Fuller test, parameters for Dewas district for ARIMA are p=1, d=1, q=5

Train ARIMA(1,1,5) for given time series and receive ARIMA model parameters to use them later as features:

	Ar,L1	Ma,L1	Ma,L2	Ma,L3	Ma,L4	Ma,L5	sigma2
72001	0,964	0,493	0,641	0,530	0,592	0,542	0,000
72002	0,837	0,351	0,578	0,555	0,563	0,521	0,004
…	…	…	…	…	…	…	…
72375	0,647	0,282	0,629	0,851	0,946	0,852	0,001

For Dewas, we have 375 collected time sirieses

ARIMA data preparation

Repeat ARIMA training for each time series

gfid	№	NDVI
72001	1	0,186
72001	2	0,184
72001	3	0,183
72001	…	…
72001	200	0,185
72002	1	0,193
72002	2	0,192
72002	3	0,195
72002	…	…
72002	200	0,197

Based on visual PAC and AC analysis and Dickey-Fuller test, parameters for Dewas district for ARIMA are p=1, d=1, q=5

	Ar,L1	Ma,L1	Ma,L2	Ma,L3	Ma,L4	Ma,L5	sigma2
72001	0,964	0,493	0,641	0,530	0,592	0,542	0,000
72002	0,837	0,351	0,578	0,555	0,563	0,521	0,004
…	…	…	…	…	…	…	…
72375	0,647	0,282	0,629	0,851	0,946	0,852	0,001

For Dewas, we have 375 collected time sireses

Classical ML+ ARIMA

Repeat ARIMA training for each time series for each district

- Karnal: ARIMA(1,0,5)
- Kaithall: ARIMA(1,1,5)
- Dewas: ARIMA(5,1,5)

On ARIMA datasets train:

- CatBoost
- LightGBM
- XGBoost
- RandomForest
- LogReg
- SVM

Compare results

Classical ML Results

Dewas

	F1-score	AUC-ROC
Logistic Regression	0,56	0,56
SVM	0,56	0,56
Random Forest	0,4	0,4
CatBoost	0,49	0,49
LightGBM	0,49	0,49
XGBoost	0,48	0,48

Karnal

	F1-score	AUC-ROC
Logistic Regression	0,52	0,5
SVM	0,49	0,46
Random Forest	0,56	0,52
CatBoost	0,51	0,49
LightGBM	0,54	0,49
XGBoost	0,54	0,5

Kaithal

	F1-score	AUC-ROC
Logistic Regression	0,42	0,42
SVM	0,38	0,38
Random Forest	0,5	0,49
CatBoost	0,5	0,49
LightGBM	0,44	0,43
XGBoost	0,54	0,53

CNN

Data preparation – Step 1, Add averaged NDVIs and drop NaNs created by averaging (now, start from 10ths observation)

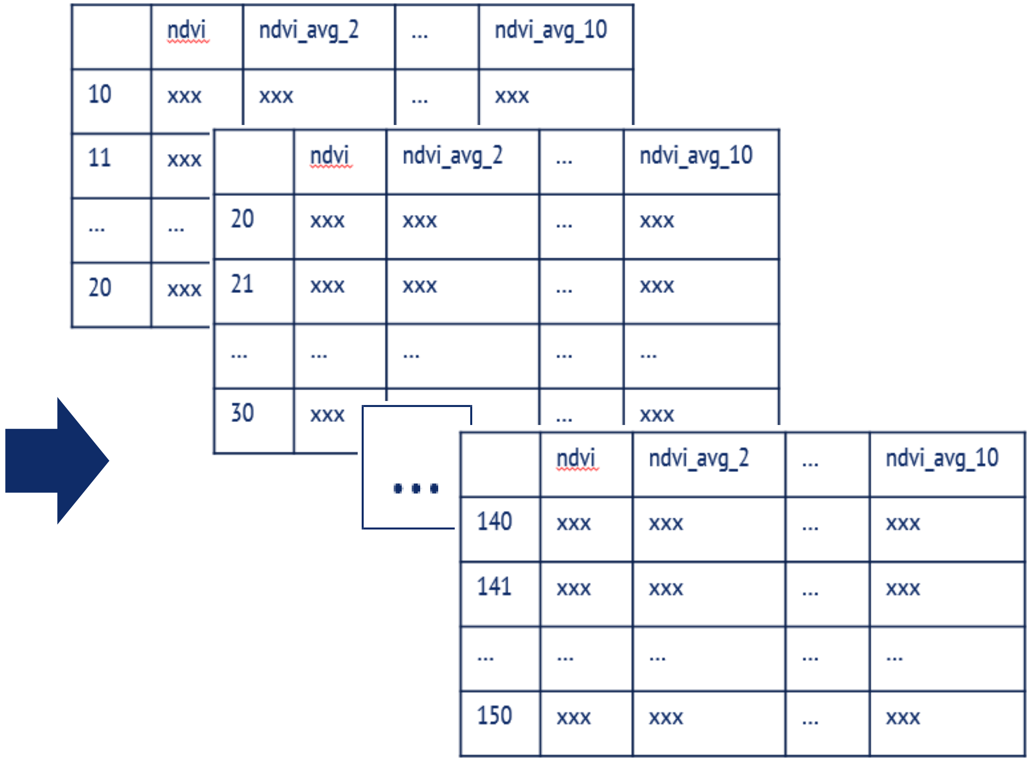
	ndvi
1	0,186
2	0,184
3	0,183
…	…
200	0,156

#	ndvi	ndvi_avg_2	ndvi_avg_3	…	ndvi_avg_8	ndvi_avg_9	ndvi_avg_10
10	0,184	0,184	0,185	…	0,196	0,201	0,206
11	0,185	0,185	0,184	…	0,192	0,195	0,199
12	0,187	0,186	0,185	…	0,189	0,191	0,194
13	0,189	0,188	0,187	…	0,187	0,189	0,191
14	0,191	0,190	0,189	…	0,187	0,188	0,189
…	…	…	…	…	…	…	…
196	0,158	0,157	0,156	…	0,152	0,151	0,151
197	0,159	0,159	0,158	…	0,153	0,153	0,152
198	0,158	0,159	0,158	…	0,155	0,154	0,153
199	0,157	0,158	0,158	…	0,156	0,155	0,154
200	0,156	0,157	0,157	…	0,156	0,156	0,155

CNN

Data preparation – Step 2, Transform dataset from 2d to 3d: crop first 150 observations into 15 smaller datasets 10x10

#	ndvi	ndvi_avg_2	ndvi_avg_3	...	ndvi_avg_8	ndvi_avg_9	ndvi_avg_10
10	0,184	0,184	0,185	...	0,196	0,201	0,206
11	0,185	0,185	0,184	...	0,192	0,195	0,199
12	0,187	0,186	0,185	...	0,189	0,191	0,194
13	0,189	0,188	0,187	...	0,187	0,189	0,191
14	0,191	0,190	0,189	...	0,187	0,188	0,189
...	...	...	...	...	...	...	...
196	0,158	0,157	0,156	...	0,152	0,151	0,151
197	0,159	0,159	0,158	...	0,153	0,153	0,152
198	0,158	0,159	0,158	...	0,155	0,154	0,153
199	0,157	0,158	0,158	...	0,156	0,155	0,154
200	0,156	0,157	0,157	...	0,156	0,156	0,155



CNN

Data preparation – Finally: for each id we have 15 datasets,
So, the final dataset size is n x 15 x 10 x 10, where n is a batch size.

	ndvi	ndvi_avg_2	...	ndvi_avg_10
10	xxx	xxx	...	xxx
11	xxx			
...	...			
20	xxx			

	ndvi	ndvi_avg_2	...	ndvi_avg_10
20	xxx	xxx	...	xxx
21	xxx	xxx	...	xxx
...	...	...	...	...
30	xxx		...	xxx

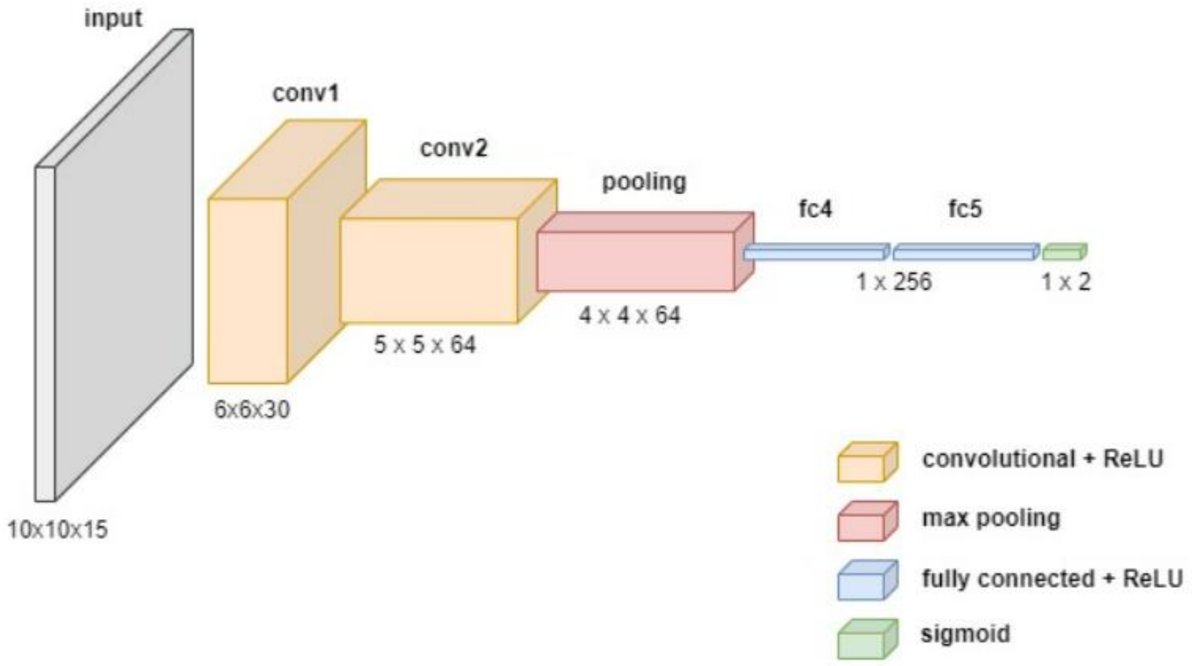
...

	ndvi	ndvi_avg_2	...	ndvi_avg_10
140	xxx	xxx	...	xxx
141	xxx	xxx	...	xxx
...	...	...	...	...
150	xxx	xxx	...	xxx

From size: (n*200) x 1
To size: **n** x 15 x 10 x 10

where **n** is number of ids
for training (or batch size)

CNN architecture



Deep Learning Results

Dewas

	F1-score	AUC-ROC
LSTM	0,42	0,47
CNN	0,8	0,8

Karnal

	F1-score	AUC-ROC
LSTM	0,61	0,56
CNN	0,76	0,74

Kaithal

	F1-score	AUC-ROC
LSTM	0,42	0,57
CNN	0,56	0,55

Outcomes

CNN is the best solution because:

- F1-score is 15-30% higher for the CNN
- Recall does not vary heavily for different classes
- Lack of time-extensive feature engineering step for CNN like ARIMA for classic ML
- Lower deviation in results for different districts

- The partner «2050, DIGITAL» is interested in non-standard feature engineering technique like ARIMA modelling
- The CNN metric results exceeded the partner's expectations

Further steps

Business:

- Add business logic to the model to create an application
- Wrap the code with any container like Docker, orchester with K8s
- Add web interface for users
- Add train automatizing flow like AirFlow or KubeFlow

Research:

- Tune hypereparameters for the CNN individually for each district
- Add ARIMA features to CNN

Thank you for you attention!