National College of Ireland

Enhancing Crop Irrigation Prediction with Transformers Model in Smart Agriculture

MSc Research Project
Msc in Data Analytics

Venkata Krishna Reddy Modem
Student ID: X23379278

School of Computing
National College of Ireland

Supervisor: Dr. Christian Horn

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Venkata Krishna Reddy Modem
Student ID:	X23379278
Programme:	Msc in Data Analytics **Year:** 2025 - 2026
Module:	Research Practicum Part-2
Supervisor:	Dr.Christian Horn
Submission Due Date:	28-01-2026
Project Title:	**Enhancing Crop Irrigation Prediction with Transformers Model in Smart Agriculture**
Word Count:	**879** **Page Count:** **06**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: M.V.K.Reddy

Date: 28-01-2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Enhancing Crop Irrigation Prediction with Transformers Model in Smart Agriculture

Venkata Krishna Reddy Modem
Student ID: x23379278
X23379278@student.ncirl.ie

1. Introduction

This project builds a predictive pipeline for smart irrigation, aiming to classify whether irrigation is required using structured agricultural data such as soil and crop attributes, moisture index, temperature, and humidity. The task is challenging because irrigation need depends on nonlinear, context-dependent interactions among heterogeneous features, and real-world sensor inputs may be noisy or inconsistent. This is addressed by the system through a progressive workflow starting with classical machine learning baseline to deep learning and ultimately a tabular transformer (FT-Transformer), which allows sound learning of feature interactions and stable model comparison to be deployed in decision support.

2. System Specifications

- **Hardware requirements:** The current multi-core CPU, optional NVIDIA graphics card with deep learning and FT-Transformer acceleration (CUDA-capable), and at least 8 GB of RAM. At least 10 GB of free space is suggested to be able to store datasets, checkpoints and experiment artifacts.

- **Software requirements:** A 64-bit operating system (Windows 10/11, Ubuntu 20.04+ or equivalent), Python 3.9+ (Python 3.10 recommended for compatibility), and an isolated environment via Conda or venv. If GPU training is used, a compatible NVIDIA driver and CUDA toolkit with cuDNN aligned to the selected deep learning framework version are required.

3. Required Libraries

Its implementation is based on a standard scientific Python stack of data processing, modeling and analysis. NumPy and Pandas are normally used to carry out data ingestion and transformation to guarantee effective numerical operations, column-based preprocessing and reproducible feature-target separation. Scikit-learn offers classical estimators of ML (e.g., Logistic Regression, KNN, Decision Tree, Gradient Boosting), powerful splitting functions, scaling functions, and coherent assessment functions. The deep learning can be performed with the help of TensorFlow/Keras as the DNN baseline and PyTorch as the FT-Transformer implementation as it is flexible in terms of custom tokenization and attention-aware

architectures. Matplotlib is used to produce visualizations (EDA curves, tuning plots and training dynamics).

Installation can be performed using the following commands:

a) **pip install numpy==2.3.5 pandas==2.3.3 scikit-learn==1.7.2 matplotlib==3.10.7 seaborn==0.13.2**
b) **pip install tensorflow==2.20.0**
c) **pip install torch==2.9.1 torchvision==0.24.1 torchaudio==2.9.1**

4. Dataset description

The dataset that was used in the current study is the structured agricultural observations of wheat crops grown on black soil in the seedling or germination stage. The records of each record include the necessary environmental and soil parameters such as moisture index (MOI), temperature, and humidity, as well as a binary label of whether irrigation is needed. All these characteristics are used as natural fluctuations in the soil moisture conditions and in atmospheric conditions that affect water demand of crops at an early stage. The dataset is realistic field pattern with moisture content slowly decreasing as temperature and humidity vary to allow the model to learn the underlying relationships that dictate the irrigation decisions. This structured table data is especially favorable to FT-Transformer model, which is effective in detecting complicated, non linear relationships among heterogeneous agricultural attributes. In spite of the fact that the sample shows the data of only one crop and soil type, the method can be scaled and modified to multi-crop, multi-soil, and multi-stage datasets to facilitate more general applications in precision agriculture.

5. Models Implemented

The project assumes a logical development of the baseline to advanced architectures to make a fair comparison and be able to measure the improvement. It starts with classical machine learning Logistic Regression a linear, interpretable model of local similarity, K-Nearest Neighbors to model local similarity, Decision Tree to model nonlinear splits, and Gradient Boosting to model local similarity with an ensemble of weak learners to improve performance. This is followed by the implementation of a deep neural network (DNN) as the main deep-learning baseline, which utilizes numerous dense layers with regularization to be learned to extract higher order nonlinear relationships among the input tabular data. Lastly, the advanced tabular architecture is an FT-Transformer. It is also effective at learning feature-to-feature interactions, through the use of self-attention to tokenize features as embeddings and thereby learns features-to-features more effectively than fixed-form models, which is well-suited to the context-dependent interaction of moisture, temperature, humidity, crop, and soil factors in irrigation prediction.

6. Code snippets and plots:

1) Data First 5 Rows

```
# Load the CSV file
df = pd.read_csv(file_path)

# first few rows
df.head()
```

	crop ID	soil_type	Seedling Stage	MOI	temp	humidity	result
0	Wheat	Black Soil	Germination	1	25	80.0	1
1	Wheat	Black Soil	Germination	2	26	77.0	1
2	Wheat	Black Soil	Germination	3	27	74.0	1
3	Wheat	Black Soil	Germination	4	28	71.0	1
4	Wheat	Black Soil	Germination	5	29	68.0	1

2) Data Info

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16411 entries, 0 to 16410
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   crop ID         16411 non-null  object
 1   soil_type       16411 non-null  object
 2   Seedling Stage  16411 non-null  object
 3   MOI             16411 non-null  int64
 4   temp            16411 non-null  int64
 5   humidity        16411 non-null  float64
 6   result          16411 non-null  int64
dtypes: float64(1), int64(3), object(3)
memory usage: 897.6+ KB
```

3) Label Encoder

```
from sklearn.preprocessing import LabelEncoder

# Create a LabelEncoder object
le = LabelEncoder()

# List of columns to encode
cols_to_encode = ['crop ID', 'soil_type', 'Seedling Stage']

# Apply label encoding to each column
for col in cols_to_encode:
    df[col] = le.fit_transform(df[col])
```

4) Missing Values

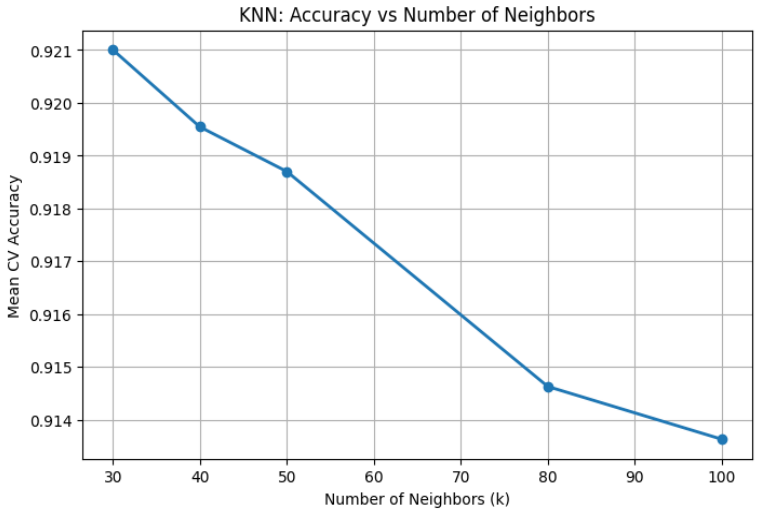
```
print(df.isnull().sum())
```

```
crop ID            0
soil_type          0
Seedling Stage     0
MOI                0
temp               0
humidity           0
result             0
dtype: int64
```

5) Splitting data

```
from sklearn.model_selection import train_test_split

#  random seed for reproducibility
seed = 42
np.random.seed(seed)

#  target variable
y = df['result']
X = df.drop('result', axis=1)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=seed)
```

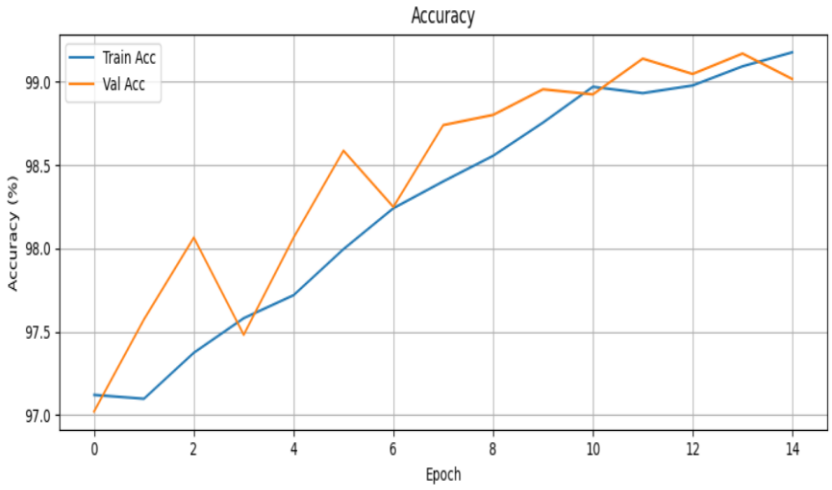
6) Hyperparameter Tuning Of KNN



7) FT-Transformer

```python
# Enhanced FT-Transformer Model
class FTTransformer(nn.Module):
    def __init__(self, num_features, num_classes, d_token=256, n_blocks=6,
                 attention_heads=8, attention_dropout=0.1, ffn_dropout=0.1,
                 ffn_factor=2):
        super().__init__()

        self.num_features = num_features
        self.d_token = d_token

        # Feature Tokenizer
        self.feature_tokenizer = NumericalFeatureTokenizer(num_features, d_token)

        # CLS token
        self.cls_token = nn.Parameter(torch.randn(1, 1, d_token) * 0.02)

        # Positional embeddings for features
        self.feature_embeddings = nn.Parameter(torch.randn(num_features, d_token) * 0.02)

        # Transformer Encoder Layers with proper configuration
        encoder_layers = []
        for _ in range(n_blocks):
            encoder_layer = nn.TransformerEncoderLayer(
                d_model=d_token,
                nhead=attention_heads,
                dim_feedforward=int(d_token * ffn_factor),
                dropout=ffn_dropout,
                activation='gelu',
                batch_first=True,
                norm_first=True)
            encoder_layers.append(encoder_layer)

        self.transformer_encoder = nn.ModuleList(encoder_layers)

        # Final Layer Normalization
        self.final_norm = nn.LayerNorm(d_token)

        # Enhanced output head with residual connection
        self.output_head = nn.Sequential(
            nn.Linear(d_token, d_token * 2),
            nn.GELU(),
            nn.Dropout(ffn_dropout),
            nn.Linear(d_token * 2, d_token),
            nn.GELU(),
            nn.Dropout(ffn_dropout),
            nn.Linear(d_token, num_classes)
        )
```

8) Accuracy plot of FT-Transformer



9) Comparison of evaluation metrics

Table 1. Performance Comparison of All Models

Model	Accuracy	Pression	Recall	F1-Score
Logistic Regression	0.8689	0.8410	0.8673	0.8539
KNN	0.9288	0.9235	0.9145	0.9190
Decision Tree	0.9414	0.9470	0.9187	0.9413
Gradient Boosting	0.9383	0.8903	0.9812	0.9336
DNN	0.9447	0.9112	0.9694	0.9394
FT-Transformer	**0.9819**	**0.9819**	**0.9819**	**0.9819**

7. Conclusion:

It presents a replicable end-to-end smart irrigation prediction pipeline, including coherent preprocessing, baseline training, deep-learning experimentation and advanced tabular modelling with standardised evaluation. The FT-Transformer was the best among the adopted methods since its attention mechanism shows the complex interactions of features in tabular agricultural data better than the classical models and a conventional DNN. It can be extended to other crops, soil, and sensor inputs, and reused in the future to support improved workflows (e.g. larger datasets, more robust validation protocols, and inference integration for deployments).

References:

1. https://scikit-learn.org/stable/
2. https://numpy.org/doc/
3. https://pandas.pydata.org/docs/
4. https://matplotlib.org/stable/index.html
5. https://www.tensorflow.org/api_docs/python/tf/keras
6. https://docs.pytorch.org/docs/stable/index.html