COMP 495 WEATHER PREDICTION PROJECT

LITERATURE REVIEW

Weather prediction plays a crucial role in our daily lives, influencing decisions related to travel plans, outdoor activities, and safety measures. Traditional methods of weather forecasting have relied on meteorological models and historical data analysis. However, advancements in machine learning techniques have opened new avenues for accurate and timely weather predictions.

This literature review explores existing research and approaches in weather prediction using machine learning, focusing on the City of Calgary as the target area.

1. Weather Prediction and Machine Learning: Numerous studies have shown the potential of machine learning algorithms in weather prediction. By analyzing historical weather data and incorporating various factors such as precipitation, sky conditions, temperature, and rain, machine learning models can learn patterns and relationships to make accurate predictions. The use of Python programming language and libraries such as NumPy, Pandas, Seaborn, and Matplotlib provides a robust framework for data analysis and visualization.

2. Data Collection and Preprocessing: Collecting accurate and comprehensive weather data is crucial for training and validating machine learning models. This can be achieved through various sources such as meteorological databases, weather APIs, or government weather stations. The collected data should be preprocessed to handle missing values, outliers, and inconsistencies, ensuring the quality and integrity of the dataset.

3. Feature Selection and Engineering: Identifying relevant features that contribute to weather patterns is essential for effective model training. Feature selection techniques, such as correlation analysis and domain knowledge, can help identify the most influential factors. Additionally, feature engineering techniques can be applied to transform and derive new features that enhance the predictive capabilities of the model.

4. Machine Learning Algorithms: Various machine learning algorithms have been explored for weather prediction, including regression models (linear regression), decision trees, random forests, XGBoost, time series models (ARIMA, LSTM), and ensemble methods. Random Forest is an ensemble algorithm that combines multiple decision trees to make predictions, offering robustness and handling non-linear relationships. XGBoost is a gradient boosting algorithm known for its powerful predictive capabilities and efficient computation. The choice of algorithm depends on the specific requirements of the project, and Scikit-learn provides a comprehensive suite of tools for training and evaluating these algorithms.

5. Model Evaluation and Performance Metrics: Accurate evaluation of the trained models is crucial to assess their performance and reliability. Commonly used metrics include mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and coefficient of determination (R-squared). Cross-validation techniques, such as k-fold cross-validation, can provide a robust estimation of the model's performance.

6. Future Directions and Challenges: While machine learning has shown promise in weather prediction, there are several challenges that researchers continue to address. These include the need for more comprehensive and high-resolution datasets, handling uncertainties and variability in weather patterns, and improving the interpretability of machine learning models. Additionally, advancements in deep learning techniques and the integration of real-time data sources can further enhance the accuracy and timeliness of weather predictions.

Weather prediction using machine learning techniques offers a promising avenue for accurate and timely forecasts. By leveraging historical weather data, feature selection, and appropriate machine learning algorithms, researchers can develop models that provide valuable insights into weather patterns and facilitate informed decision-making. The use of Python programming language and libraries, along with machine learning frameworks like Scikit-learn, provides a robust framework for weather analysis and prediction. Future research in this field holds the potential to further improve the accuracy and reliability of weather forecasting, benefiting individuals and communities in their daily activities and safety measures.

PROJECT SCOPE

This project aims to predict weather conditions for the City of Calgary using machine learning techniques. By leveraging historical weather data, including factors such as temperature, relative humidity, surface pressure, apparent temperature, cloud cover, precipitation, snowfall, and wind speed, a machine learning model will be developed to forecast weather patterns for a given week. The project builds upon the previous work in COMP 495, where historical weather data was imported from an API in CSV format.

Focus/Content:

This project extends the previous work in COMP 495 by focusing on weather prediction for the City of Calgary. By leveraging historical weather data, Azure database, and machine learning algorithms, the project aims to develop an accurate and reliable model for forecasting weather conditions. The dashboard allows users to fetch weather data for the next 7 days and view the predicted temperature and weather conditions. The project report will provide insights into the performance of the model, allowing for informed decision-making and preparedness for weather-related scenarios in Calgary. Below are the main steps involved in this project:

1. Data Collection and Database Setup: Create an Azure database and set up a table to store the weather data. Fetch feature data from an API for the next 7 days and store it in the Azure database.
2. Data Preparation: Clean and preprocess the imported weather data, handling missing values, outliers, and inconsistencies to ensure data quality.
3. Feature Selection: Identify relevant weather factors such as temperature, humidity, pressure, cloud cover, precipitation, snowfall, and wind speed as predictors for the machine learning model.

4. Model Training and Evaluation: Utilize machine learning algorithms such as regression models, decision trees, random forests, and XGBoost to train a predictive model using the historical weather data. Split the dataset into training and testing sets to evaluate the performance of the trained model.
5. Fine-tune the model by adjusting hyperparameters and applying feature engineering techniques to improve its predictive capabilities.
6. Model Prediction and Database Integration: Create a dashboard using pyodbc to allow users to interact with the weather prediction system. When the user clicks on the "Fetch Data" button for the next 7 days, retrieve the feature data from the Azure database. Feed this data from the database to the trained machine learning model to produce temperature and weather condition predictions for the next 7 days. Store the predictions back in the Azure database for display and future reference.
7. Visualization and Reporting: Create visualizations on the dashboard using Python libraries like pyodbc, Matplotlib and Seaborn to present the predicted weather patterns for the next 7 days. Display the predicted temperature and weather conditions on the dashboard, allowing users to visualize the forecasted data in an interactive and user-friendly manner.

PROJECT DESIGN AND TECHNICAL STRATEGY:

1. Data Collection and Database Setup:

    - Create an Azure database and set up a table for storing weather data.
    - Fetch feature data from an API for the next 7 days and store it in the Azure database.

2. Data Preparation:

    - This step is covered in COMP 495 and will use the CSV data that was imported and preprocessed.
    - Clean and preprocess the imported weather data, handling missing values, outliers, and inconsistencies.
    - Data will be split into a set of training and testing data (80%, 20%)so that it can be fed to the machine learning model of choice.

3. Feature Selection:
    - Identify relevant weather factors (temperature, humidity, pressure, cloud cover, precipitation, snowfall, wind speed) as predictors for the machine learning model.

4. Model Training and Evaluation:

    - Utilize machine learning algorithms such as regression models (e.g., Linear Regression), ensemble methods (e.g., Random Forest), and gradient boosting (e.g., XGBoost) to train a predictive model using historical weather data.
    - Split the dataset into training and testing sets, ensuring a representative distribution of data for evaluation.

- Perform model evaluation by measuring key metrics such as mean squared error (MSE), root mean squared error (RMSE), and coefficient of determination (R-squared).
- Compare the performance of the different models (Linear Regression, XGBoost, Random Forest) based on the evaluation metrics to determine the most effective algorithm for weather prediction.

5. Model Prediction and Database Integration:
    - Create a dashboard using pyodbc to enable user interaction with the weather prediction system.
    - Retrieve feature data from the Azure database when the user clicks on the "Fetch Data" button.
    - Feed the data into the trained machine learning model to generate temperature and weather condition predictions for the next 7 days.
    - Store the predictions back in the Azure database for display and future reference.

6. Visualization and Reporting (User Interface):
    - Utilize Python libraries (pyodbc, Matplotlib, Seaborn) to create visualizations on the dashboard.
    - Display the predicted temperature and weather conditions in an interactive and user-friendly manner.

7. Project Report:
    - Generate a comprehensive project report documenting the methodology, findings, and limitations of the weather prediction model.
    - Include insights into the model's performance and recommendations for improvement.

The project aims to develop an accurate weather prediction model for the City of Calgary using historical data, machine learning algorithms, and Azure database integration. The dashboard allows users to fetch and visualize the predicted weather patterns for the next 7 days.

The project design emphasizes data preparation, feature selection, model training, and evaluation, as well as the integration of the prediction system with the dashboard. The project report provides a detailed analysis of the model's performance, ensuring informed decision-making and preparedness for weather-related scenarios in Calgary.

Project Look and Feel:

The project's look and feel will revolve around an interactive dashboard created using the pyodbc Python library. The dashboard serves as the central hub, connecting the backend Azure Database and the machine learning model.

Upon accessing the dashboard, users will have the ability to fetch weather data for the next 7 days by clicking the fetch button. Behind the scenes, the dashboard interacts with the Azure Database, pulling the necessary feature data from the Open-Meteo API and storing it in the database for processing.

The dashboard will provide a user-friendly interface, presenting the predicted weather patterns and conditions for the next 7 days. Utilizing Python libraries such as Matplotlib and Seaborn, the dashboard will showcase visually appealing and informative visualizations, enabling users to explore the forecasted data effortlessly.

Through the integration of the backend Azure Database, the dashboard seamlessly connects with the machine learning model. The fetched data will be fed into the model, generating predictions for temperature and weather conditions. These predictions will be stored back in the Azure Database, allowing the dashboard to display the forecasted data alongside the actual weather data.

Overall, the project's look and feel will emphasize the convenience and accessibility of the dashboard, empowering users to make informed decisions and gain valuable insights into the future weather conditions of the City of Calgary.

8 WEEK MILESTONES WITH TIMELINE:

Week 1: Project Setup and Data Collection

- Set up the development environment and install necessary libraries.
- Create an Azure database and configure the required tables.
- Fetch historical weather data for the City of Calgary from the Open-Meteo API (Get from COMP 495)
- Import and preprocess the data, ensuring data quality and consistency (Covered in COMP 495)

Week 2: Feature Selection and Preparation

- Identify relevant weather factors for prediction, such as temperature, humidity, pressure, cloud cover, precipitation, snowfall, and wind speed.
- Perform feature engineering and selection to enhance the predictive capabilities of the model.
- Split the dataset into training and testing sets for evaluation.

Week 3: Model Training and Evaluation

- Implement machine learning algorithms including regression models (linear regression), ensemble methods (random forests), and XGBoost.
- Train the models using the training dataset and fine-tune the models by adjusting hyperparameters.
- Evaluate the performance of each model using appropriate evaluation metrics.

- Select the model with the best performance and used that for the project

Week 4: Dashboard Development (Frontend)

- Design and develop an interactive dashboard using the pyodbc Python library.
- Create a user-friendly interface with buttons and visual elements to fetch and display weather data.
- Integrate the necessary functionalities to connect with the backend Azure Database.

Week 5: Backend Integration and Database Management

- Establish the connection between the dashboard and the Azure Database using pyodbc.
- Develop the functionality to fetch feature data from the API for the next 7 days.
- Store the fetched data into the Azure Database for future processing and retrieval.

Week 6: Model Integration and Prediction

- Implement the integration of the trained machine learning models with the dashboard.
- Develop the backend process to feed the fetched data into the models for prediction.
- Store the predicted temperature and weather conditions back into the Azure Database.

Week 7: Visualization and Reporting

- Utilize Python libraries like Matplotlib and Seaborn to create visualizations of the predicted weather patterns on the dashboard.
- Display the forecasted data alongside the actual weather data for easy comparison.
- Generate a comprehensive project report documenting the methodology, findings, and limitations of the weather prediction model.

Week 8: Testing, Refinement, and Finalization

- Conduct thorough testing of the dashboard functionalities, ensuring proper data fetching, prediction, and visualization.
- Refine and optimize the codebase, improving performance and addressing any bugs or issues.
- Finalize the project documentation, including the project report.

PROJECT DELIVERABLES:

1. Azure database setup and configured for storing weather data.
2. Cleaned and preprocessed historical weather data from the Open-Meteo API.
3. Feature selection and preparation, including relevant weather factors for prediction.
4. Trained machine learning models using regression, random forests, and XGBoost algorithms.
5. Evaluation of model performance using appropriate metrics and testing on the testing dataset.
6. Development of an interactive dashboard using the pyodbc Python library.
7. Integration of the dashboard with the Azure Database for data retrieval and storage.
8. Backend process to fetch feature data from the API and feed it into the models for prediction.
9. Display of forecasted temperature and weather conditions on the dashboard.
10. Comprehensive project report documenting methodology, findings, and limitations of the model.
11. Finalized codebase with optimized performance, bug fixes, and refinements.
12. Complete project documentation, including the project report and any additional required documentation.

EVALUATION CRITERIA

The evaluation of the Weather Analysis project will be based on the completeness and quality of the project deliverables, including the Jupyter Notebook, dashboard, and project report. The evaluation criteria are as follows:

Project Proposal (15%):

- Alignment with Objectives: Assessing the consistency between the proposed objectives, methodologies, and deliverables outlined in the project proposal and their actual implementation.
- Feasibility: Evaluating the practicality of the project proposal, considering the availability of resources and time constraints.

Python Code and Dashboard (50%):

- Code Quality: Evaluating the organization, structure, readability, and adherence to best coding practices in the Python code implementation.
- Data Import and Preprocessing: Assessing the proper handling of data import, preprocessing, and cleaning techniques in the code.

- Analysis Operations: Evaluating the implementation of data analysis operations using appropriate libraries and techniques, such as NumPy, Pandas, and Scikit-learn.
- Interactive Dashboard: Assessing the effectiveness and functionality of the dashboard, including its integration with the Azure Database, user-friendly interface, and visualization of the predicted weather patterns.

Project Report (35%):

- Structure and Organization: Evaluating the logical flow and organization of the project report, including clear sections, subheadings, and a comprehensive table of contents.
- Clarity of Explanations: Assessing the clarity and conciseness of the project objectives, methodologies, and findings explained in the report.
- Interpretation of Results: Evaluating the depth of analysis and interpretation of the weather data, supported by appropriate visualizations and statistical measures.
- Completeness: Ensuring the inclusion of all relevant information, including data sources, preprocessing steps, analysis techniques, model evaluation metrics, and limitations of the project.