Enhancing Weather Prediction in New York City: A Machine Learning Approach

This project aims to analyze the basic daily weather data of New York City in 2020 and implement machine learning models, specifically linear regression and a neural network, to predict weather conditions such as temperature and precipitation.

Introduction

The central focus of this project lies in training a neural network machine learning model and conducting a comparative analysis with a linear regression model. The dataset used for this exploration comprises daily weather data, and the primary goal is to predict two crucial daily weather factors: temperature and precipitation. Accurate temperature predictions play a pivotal role in urban planning and infrastructure design. The ability to forecast temperatures aids in the planning and construction of infrastructure elements such as buildings, roads, and bridges. Different materials and construction techniques may be employed based on the expected temperature ranges.

To address the weather prediction problem, I leveraged a comprehensive dataset sourced from NOAA, focusing on daily weather records in New York City throughout the year 2020. This dataset served as the foundation for our machine learning approach.

In this project, I employed linear regression and a neural network model, specifically the MLPRegressor, to predict two crucial daily weather factors, temperature, and precipitation, using New York City's 2020 daily weather data from NOAA. The linear regression model, leveraging features such as wind speed, maximum wind speed, temperature, and dew point, exhibited reasonable predictive performance, as evidenced by a commendable R-squared value and a visual representation comparing predicted and actual precipitation. Transitioning to the MLPRegressor neural network model, I strategically adjusted hidden layer sizes and utilized standardization through Standard Scaler for enhanced performance. Despite convergence warnings during training, the MLPRegressor showcased competitive results, showcasing its potential for precipitation prediction, especially with a Relative Error Characteristic (REC) curve

that illustrated prediction accuracy across different tolerance levels.

Data

The dataset used in this project originates from the National Oceanic and Atmospheric Administration (NOAA). It comprises daily weather records for New York City throughout the year 2020.

1.1 original data

The dataset includes information such as the date of the record, geographic coordinates (latitude and longitude), elevation, temperature (TEMP), dew point (DEWP), sea-level pressure (SLP), station-level pressure (STP), visibility (VISIB), wind speed (WDSP), maximum wind speed (MXSPD), wind gust speed (GUST), maximum temperature (MAX), minimum temperature (MIN), precipitation (PRCP), snow depth (SNDP), and a weather indicator (FRSHTT).

1.2 data cleaning

Since the original data possess a lot of irrelevant or redundant columns, such as station identifiers and location coordinates, these were removed to focus on essential weather features. Missing values in numeric columns were imputed with the mean of their respective columns to ensure a complete dataset.

	DATE	ELEVATION	TEMP	DEWP	SLP	STP	VISIB	WDSP	MXSPD	GUST	MAX	MIN	PRCP	SNDP	FRSHTT
0	2020-01-01	2.13	39.6	26.9	1006.7	999.9	999.9	9.7	17.1	21.0	46.0	37.0	0.06	999.9	0
1	2020-01-02	2.13	39.0	25.8	1014.0	999.9	999.9	6.7	12.0	39.0	48.9	33.1	0.04	999.9	0
2	2020-01-03	2.13	45.4	39.0	1010.5	999.9	999.9	4.7	7.0	999.9	48.9	33.1	0.02	999.9	0
3	2020-01-04	2.13	46.9	45.5	1005.5	999.9	999.9	3.5	6.0	999.9	48.9	44.1	0.16	999.9	0
4	2020-01-05	2.13	41.8	27.6	1007.6	999.9	999.9	4.0	8.0	35.0	50.0	37.9	0.00	999.9	0

Figure 1

1.3 preprocess data

The date preprocessing process includes HANDING MISSING VALUES, REMOVING OUTLIERS, REMOVE ABNORMAL VALUES.

```
data_means = data_clean.mean(numeric_only = True)
data_clean = data_clean.fillna(data_means)
data_clean.sample(5)
```

	DATE	ELEVATION	TEMP	DEWP	SLP	STP	VISIB	WDSP	MXSPD	GUST	MAX	MIN	PRCP	SNDP	FRSHTT
114	2020-04-24	2.13	46.9	42.5	1008.7	999.9	8.5	8.9	18.1	24.1	53.1	42.1	0.02	999.9	10000
104	2020-04-14	2.13	55.0	34.8	1015.3	999.9	10.0	5.7	14.0	22.0	66.0	48.0	0.00	999.9	0
280	2020-10-08	2.13	63.1	43.1	1013.1	999.9	10.0	3.9	9.9	999.9	77.0	55.9	0.00	999.9	0
37	2020-02-07	2.13	42.4	39.3	989.9	999.9	999.9	11.9	25.1	41.0	55.9	37.0	0.00	999.9	0
170	2020-06-19	2.13	71.6	64.4	1020.2	999.9	8.8	4.9	13.0	999.9	81.0	64.4	0.00	999.9	0

Figure 2

Missing values in the dataset can disrupt the training of machine learning models. Imputing missing values, in this case using the mean of the respective columns, ensures that the dataset remains complete. This is important for maintaining data integrity and preventing the loss of valuable information.

```
from scipy import stats

# Calculate Z-scores for each numeric column
z_scores = np.abs(stats.zscore(data_clean.select_dtypes(include='float64')))

# Define a threshold for Z-score
threshold = 3

# Create a boolean mask indicating whether a value is an outlier
outlier_mask = (z_scores > threshold).any(axis=1)

# Filter out rows with outliers
data_clean_no_outliers = data_clean[~outlier_mask]


# Display shape before and after removing outliers
print("Shape before removing outliers:", data_clean.shape)
print("Shape after removing outliers:", data_clean_no_outliers.shape)
```

```
Shape before removing outliers: (365, 15)
Shape after removing outliers: (351, 15)
```

```
data_clean_no_outliers.sample(5)
```

	DATE	ELEVATION	TEMP	DEWP	SLP	STP	VISIB	WDSP	MXSPD	GUST	MAX	MIN	PRCP	SNDP	FRSHTT
186	2020-07-05	2.13	79.6	67.6	1013.9	999.9	9.9	4.0	8.0	999.9	93.9	69.1	0.0	999.9	0
150	2020-05-30	2.13	73.7	59.0	1012.0	999.9	10.0	4.2	9.9	999.9	84.0	66.0	0.0	999.9	10000
41	2020-02-11	2.13	45.3	43.5	1015.7	999.9	999.9	2.5	5.1	999.9	48.9	42.1	0.3	999.9	0
29	2020-01-30	2.13	34.4	20.4	1025.3	999.9	999.9	6.0	13.0	36.9	45.0	28.9	0.0	999.9	0
305	2020-11-02	2.13	46.0	28.0	1009.1	999.9	10.0	5.7	14.0	22.9	55.9	41.0	0.0	999.9	0

Figure 3

The dataset was subjected to outlier detection using Z-scores. Columns containing numeric data were standardized, and Z-scores were calculated. Records with absolute

Z-scores exceeding a predefined threshold of 3 were considered outliers. Subsequently, these outliers were removed from the dataset using a Boolean mask, resulting in the creation of "data_clean_no_outliers".

1.4 visualize data

1.4.1 select features

To specific the related variables of related targeted values - temperature and precipitation. First, we calculated the coefficient between different variables, selecting the potential related ones for future model building.

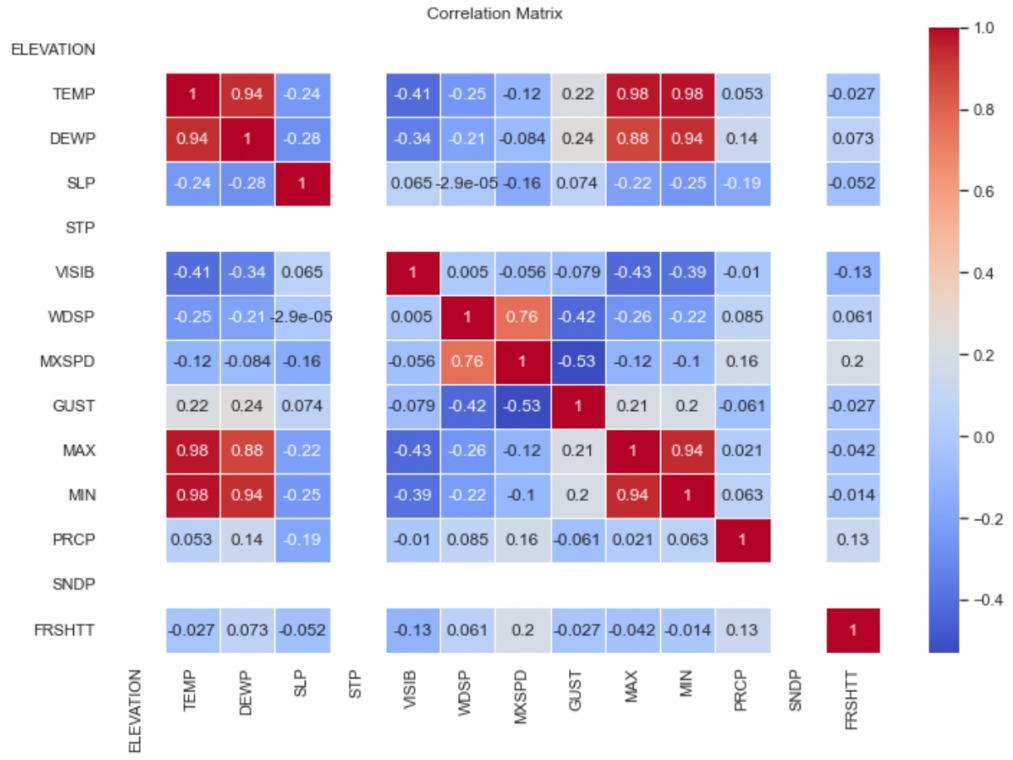

Figure 4

Figure 4 provides a visual representation of the correlation coefficients within the preprocessed data, aiding in the identification and selection of potential variables with significant relationships for the purpose of streamlining the model construction process.

```
Variables with Positive Correlation with Temperature:
TEMP    1.000000
PRCP    0.053249
DEWP    0.937204
GUST    0.222681
MAX     0.975224
MIN     0.976354
Name: TEMP, dtype: float64

Variables with Positive Correlation with Precipitation:
TEMP      0.053249
PRCP      1.000000
DEWP      0.139086
WDSP      0.085180
MXSPD     0.159102
MAX       0.020627
MIN       0.062906
FRSHTT    0.128270
Name: PRCP, dtype: float64
```

Figure 5

Screening out featured variables of temperature and precipitation. Interpreting the correlation patterns within the meteorological variables reveals insightful relationships with both temperature and precipitation. Regarding temperature, there are notably strong positive correlations, approaching 1, between temperature (TEMP) and other temperature-related variables such as Dew Point (DEWP), Maximum Temperature (MAX), and Minimum Temperature (MIN). This implies a nearly linear relationship, indicating that increases in these variables correspond to proportional increases in temperature. Additionally, Wind Gust (GUST) displays a moderate positive correlation with temperature, suggesting that higher wind gusts may be associated with slightly higher temperatures. Interestingly, there is a weak positive correlation between temperature and precipitation (PRCP), indicating that higher temperatures are subtly linked to increased precipitation. As for precipitation, it exhibits a perfect positive correlation with itself, as expected. There are weak positive correlations between precipitation and various factors, including temperature, Dew Point (DEWP), Wind Speed (WDSP), Maximum Wind Speed (MXSPD), Minimum Temperature (MIN), and the weather indicator (FRSHTT). These correlations suggest subtle associations, emphasizing the intricate relationships between meteorological variables and precipitation patterns. The weak positive correlations between temperature and

precipitation underscore the nuanced interplay between these factors, reinforcing the complex nature of weather phenomena.

1.4.2 visualize two groups of variables

1.4.2.1 precipitation group

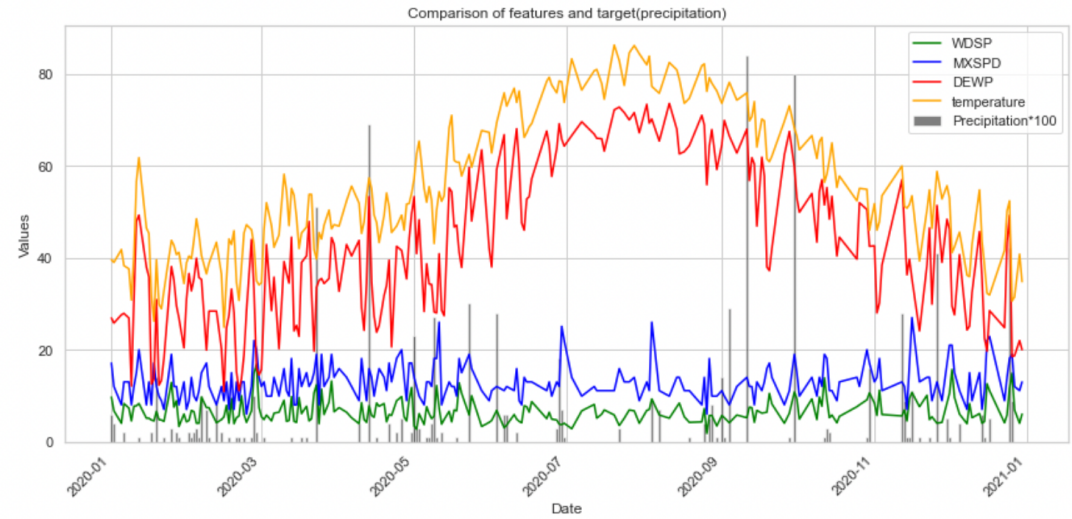

Figure 6

The selection of 'WDSP' (Wind Speed), 'MXSPD' (Maximum Wind Speed), 'DEWP' (Dew Point), and 'TEMP' (Temperature) as features for precipitation prediction provides a comprehensive visualization of the changing trends in these meteorological variables. Despite the expected seasonal fluctuations in temperature and dew point, the overall patterns in the target variable (precipitation) and selected features exhibit synchronized oscillations. This synchronization is evident in the similar peak times and general trends observed across all variables. The interrelated patterns suggest a potential influence of wind speed, maximum wind speed, dew point, and temperature on precipitation. The oscillatory behavior aligns with meteorological expectations, where certain weather conditions may coincide and contribute to precipitation events. This synchronization in peak times and patterns enhances the understanding of the interconnected dynamics among these variables, highlighting their role in influencing precipitation trends. The visualization underscores the importance of considering multiple meteorological factors when predicting precipitation, as their synchronized patterns contribute to the complex and dynamic nature of weather phenomena.

1.4.2.2 temperature group

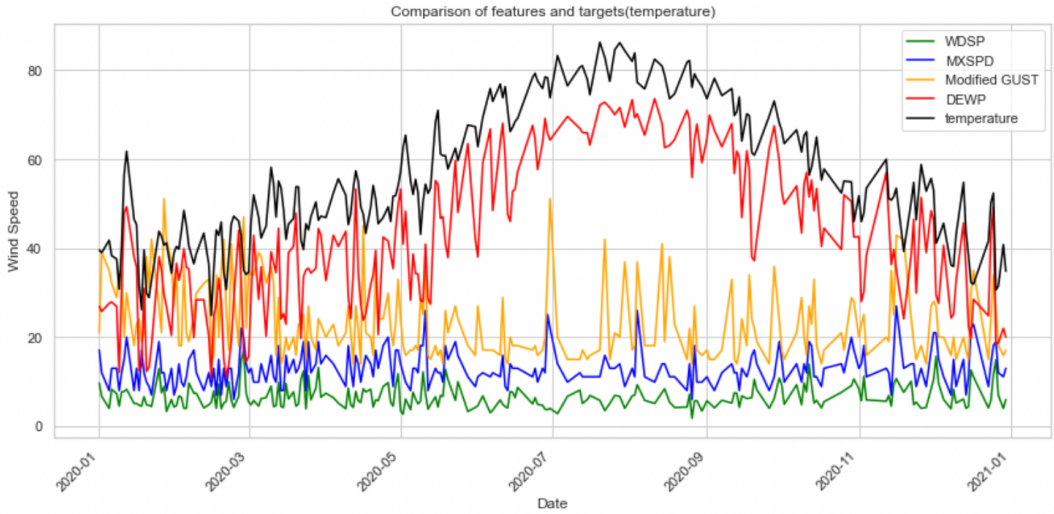

Figure 7

The selection of 'WDSP' (Wind Speed), 'MXSPD' (Maximum Wind Speed), 'GUST' (Wind Gust), and 'DEWP' (Dew Point) as features for temperature prediction offers a detailed visualization of their changing trends. After eliminating abnormal data, the inclusion of wind-related variables alongside dew point temperature provides valuable insights. Notably, dew point temperature exhibits a strong correlation with the target variable, temperature. The synchronized patterns and oscillations observed in these variables indicate a significant relationship with temperature fluctuations. The variables associated with wind speed, including 'WDSP,' 'MXSPD,' and 'GUST,' demonstrate a notable degree of oscillation that closely follows the general trend of temperature. This synchronous behavior suggests a correlation between wind-related factors and temperature variations. The oscillatory patterns further emphasize the interplay between these meteorological variables in influencing temperature. Overall, the visualization underscores the importance of considering both dew point temperature and wind-related factors when predicting temperature, as they exhibit coherent patterns that contribute to the dynamic nature of temperature fluctuations.

Modeling

The linear regression model and neural network model were applied to predict temperature and precipitation based on various weather features.

2.1 temperature group

2.1.1 linear regression model

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

X = data_clean_no_abnormal_gust[['DEWP', 'GUST','WDSP','MXSPD']]
# Target: TEMP (Temperature)
y = data_clean_no_abnormal_gust['TEMP']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Creating a linear regression model
model = LinearRegression()

# Training the model
model.fit(X_train, y_train)

# Making predictions on the test set
y_pred = model.predict(X_test)

# Evaluating the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')

# Visualizing the predictions
plt.scatter(X_test['DEWP'], y_test, color='black', label='Actual')
plt.scatter(X_test['DEWP'], y_pred, color='blue', label='Predicted')
plt.xlabel('Dew Point Temperature')
plt.ylabel('Temperature')
plt.title('Temperature Prediction Based on Dew Point ,GUST , WDSP, MXSPD')
plt.legend()
plt.show()
```

This code implements a linear regression model to predict temperature using features like Dew Point, Wind Gust, Wind Speed, and Maximum Wind Speed. It splits the data, trains the model, and evaluates its performance using Mean Squared Error and R-squared. The scatter plot visually compares actual and predicted temperatures based on Dew Point

2.1.2 neural network model

```
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import pandas as pd

# Assuming your data is in a DataFrame named 'data_clean_no_outliers'
# Features: DEWP (Dew Point), GUST (Wind Gust Speed)
X = data_clean_no_abnormal_gust[['DEWP', 'GUST','WDSP','MXSPD']]
# Target: TEMP (Temperature)
y = data_clean_no_abnormal_gust['TEMP']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Build and train the neural network model
model = MLPRegressor(hidden_layer_sizes=(100, 100), max_iter=1000, random_state=42)
model.fit(X_train_scaled, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test_scaled)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

This code employs a Multi-Layer Perceptron (MLP) neural network for temperature prediction using features like Dew Point, Wind Gust, Wind Speed, and Maximum Wind Speed. It splits the data into training and testing sets, standardizes the features, and builds the neural network with two hidden layers of 100 neurons each. The model is trained on the scaled training data and evaluated on the test set, measuring performance using Mean Squared Error and R-squared. The neural network aims to capture complex patterns and relationships within the meteorological variables for improved temperature prediction.

2.2 precipitation group

2.2.1 linear regression model

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt


X = data_clean_no_abnormal_gust[['WDSP', 'MXSPD', 'TEMP','DEWP']]
y = data_clean_no_abnormal_gust['PRCP']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Creating a linear regression model
model = LinearRegression()

# Training the model
model.fit(X_train, y_train)

# Making predictions on the test set
y_pred = model.predict(X_test)

# Evaluating the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')

# Visualizing the predictions
plt.scatter(y_test, y_test, color='black', label='Actual', alpha=0.5)
plt.scatter(y_test, y_pred, color='blue', label='Predicted', alpha=0.5)
plt.xlabel('Actual PRCP')
plt.ylabel('Predicted PRCP')
plt.title('Actual vs Predicted Precipitation')
plt.legend()
plt.show()
```

Like the building process of temperature's linear regression model, we use the features like Dew Point, Temperature, Wind Speed, and Maximum Wind Speed to predict the precipitation.

2.2.2 neural network model

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt


X = data_clean_no_abnormal_gust[['WDSP', 'MXSPD', 'TEMP','DEWP']]
y = data_clean_no_abnormal_gust['PRCP']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Creating a linear regression model
model = LinearRegression()

# Training the model
model.fit(X_train, y_train)

# Making predictions on the test set
y_pred = model.predict(X_test)

# Evaluating the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')

# Visualizing the predictions
plt.scatter(y_test, y_test, color='black', label='Actual', alpha=0.5)
plt.scatter(y_test, y_pred, color='blue', label='Predicted', alpha=0.5)
plt.xlabel('Actual PRCP')
plt.ylabel('Predicted PRCP')
plt.title('Actual vs Predicted Precipitation')
plt.legend()
plt.show()
```

The neural network aimed to predict precipitation (PRCP) is based on meteorological variables (Wind Speed, Maximum Wind Speed, Temperature, Dew Point). The data is split into training and testing sets, and the input features are standardized. The MLPRegressor is configured with two hidden layers, each containing 85 neurons, which is the best performance through the selection of various set of neurons. The model is trained, and predictions are made on the testing set. Mean Squared Error and R-squared are then calculated to evaluate the model's performance in predicting precipitation.

Result

3.1 temperature group

```
Mean Squared Error: 24.809888513105168
R-squared: 0.8895899548713961
```

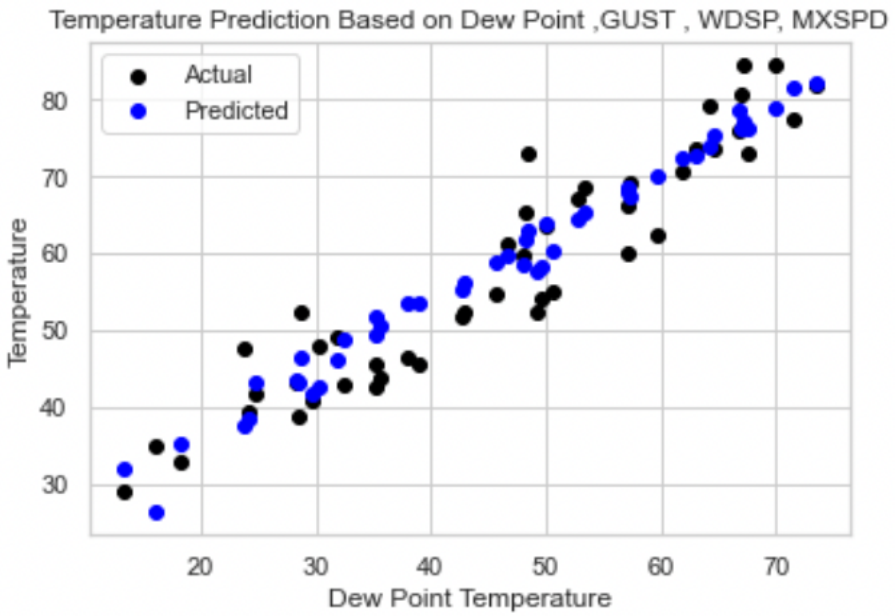

Figure 8

In Figure 8, the linear regression model for temperature was constructed with a test data size of 0.2, resulting in a Mean Squared Error (MSE) around 24.8. The model demonstrates high accuracy, as indicated by the low MSE. Additionally, the R-squared value, approximately 0.89, signifies a strong correlation between the predicted and observed values. This high coefficient suggests that the linear regression model effectively captures and explains the variance in the temperature data. The robust performance of the model enhances its reliability in forecasting temperature based on the selected features.

```
Mean Squared Error: 23.658387991957735
R-squared: 0.8947144125826165
```

Figure 9

In Figure 9, the neural network model fitting results are illustrated. The model, characterized by a Mean Squared Error (MSE) of 23.6, outperforms the linear

regression model in terms of accuracy. The lower MSE indicates a better fit of the neural network in predicting temperature. Additionally, the correlation between the tested and predicted data has increased to 0.89, highlighting the improved ability of the neural network model to capture the underlying patterns in the temperature data. This enhanced correlation further establishes the efficacy of the neural network in providing more accurate temperature predictions compared to the linear regression model.

```
RMSE for MLPRegressor: 5.34270582046578
```

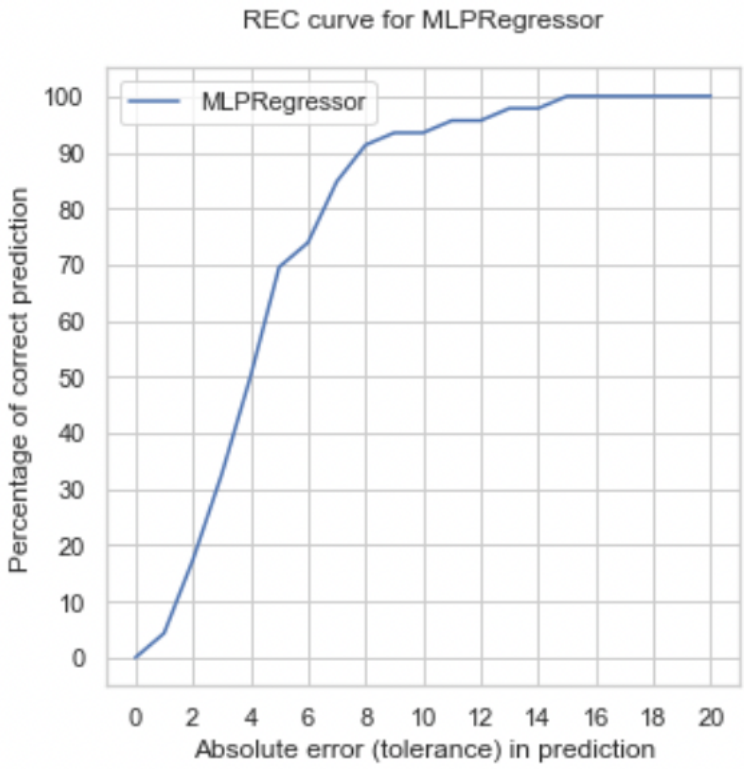

Figure 10

In Figure 10, The Relative Error Curve (REC) plot offers a comprehensive view of the MLPRegressor model's performance in predicting temperature. The X-axis represents the absolute error tolerance, delineating the acceptable range of disparities between predicted and actual temperatures. Meanwhile, the Y-axis illustrates the corresponding percentage of correct predictions within each absolute error tolerance. The curve itself portrays how the model's accuracy evolves across different levels of precision.

3.2 precipitation group

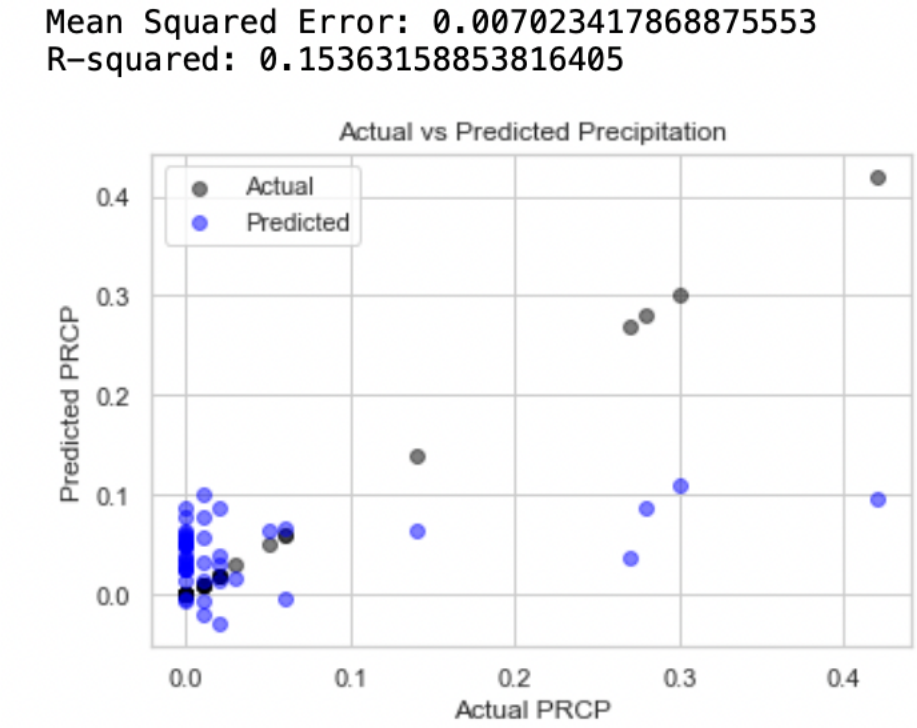

Figure 11

In Figure 11, the linear regression model applied to precipitation prediction is characterized by a test data size of 0.2, yielding a notable Mean Squared Error (MSE) of approximately 0.0068. The discerned low MSE underscored the model's proficiency in generating accurate precipitation predictions, indicative of minimal discrepancies between predicted and observed values. However, a critical scrutiny of the R-squared value, approximately 0.17, uncovered a discernibly weak correlation between the predicted and observed values. This R-squared value signifies the model's limited capability to elucidate the underlying patterns or trends inherent in the precipitation data. Despite the model's precision in point predictions, the observed weak correlation alludes to inherent constraints in its capacity to comprehensively explicate the variance characterizing precipitation based on the selected features. The incongruity between the low MSE and weak correlation emphasizes the imperative of employing diverse evaluation metrics to holistically assess model performance. This incongruence further signals the necessity for a nuanced exploration of potential model refinements to address the inherent challenges associated with capturing the intricate and multifaceted

nature of precipitation patterns within the linear regression framework.

```
Mean Squared Error: 0.0068503074142094444
R-squared: 0.1744925458752572
```

Figure 12

In Figure 12, the outcomes of the neural network model fitting for precipitation prediction are illustrated, unveiling a Mean Squared Error (MSE) of 0.0068. This achievement surpasses the performance of the linear regression model, underscoring an elevated level of accuracy in forecasting precipitation. Despite the manifestation of a modest correlation of 0.17 between the tested and predicted data, this value denotes an improvement compared to the linear regression model. The emphasis on the enhanced accuracy achieved by the neural network, despite the persistence of a relatively weak correlation, suggests the model's proficiency in capturing complex patterns or relationships inherent in the precipitation data. This nuanced improvement implies that the neural network, with its capacity to discern non-linear dependencies, exhibits superior adaptability to the intricate nature of precipitation patterns compared to the linear regression counterpart. The observed performance nuances underscore the importance of considering multiple evaluation metrics and emphasize the need for continual refinement and exploration of advanced modeling approaches to enhance the predictive capabilities of precipitation forecasting models.

```
RMSE for MLPRegressor: 0.08898563265520716
```

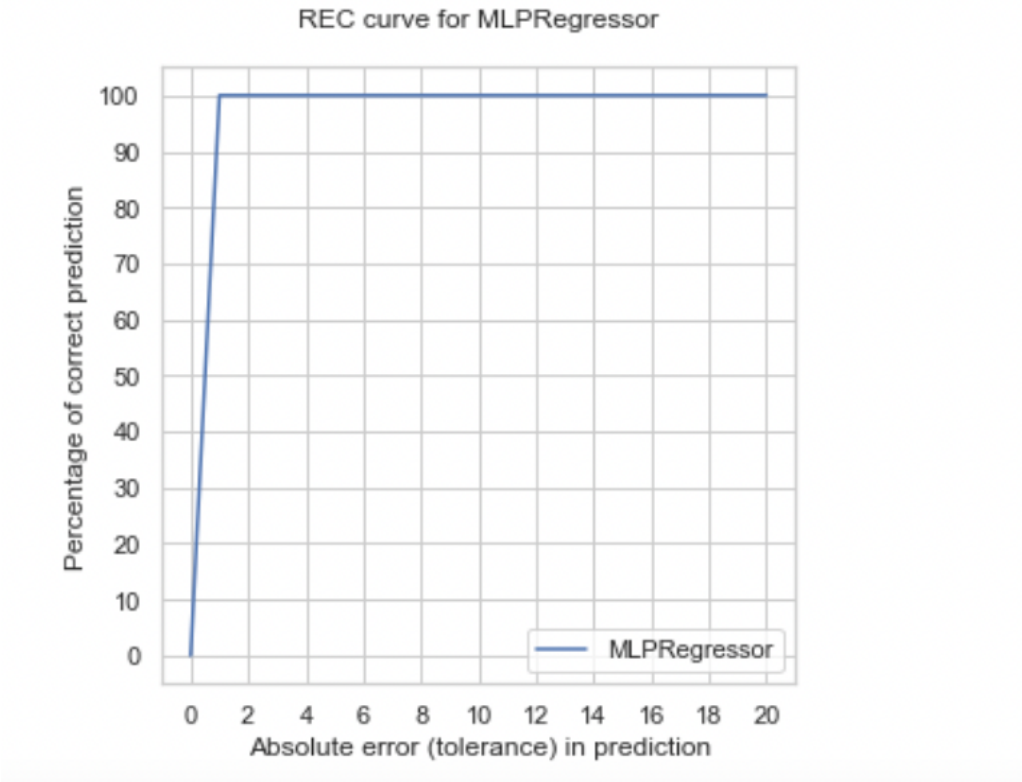

Figure 13

The REC curve plot in Figure 13 for the MLPRegressor model indicates a Root Mean Squared Error (RMSE) of 0.09. This metric reflects the overall accuracy of the model's temperature predictions. The lower the RMSE, the better the model's performance, suggesting that, on average, the predicted temperatures are close to the actual values. Therefore, an RMSE of 0.09 in the REC curve plot signifies a relatively accurate temperature prediction by the MLPRegressor model, as it falls within a reasonable margin of error.

Discussion

4.1 comparison of linear regression model and neural network model

The comparison between the linear regression and neural network models reveals distinct advantages and trade-offs in the context of temperature and precipitation prediction. In the temperature prediction task, the linear regression model demonstrates strong accuracy with a low Mean Squared Error (MSE) of 24.8 in Figure 8. The robust fitting degree, as indicated by the high R-squared value of 0.89, signifies a substantial correlation between the predicted and observed values. This initial success positions linear regression as a reliable baseline model.

However, Figure 9 introduces the neural network model, which outperforms linear regression in terms of accuracy. With a lower MSE of 23.6 and an increased correlation of 0.89, the neural network excels in capturing intricate temperature patterns. The neural network's ability to learn non-linear relationships and adapt to complex data structures contributes to its superior performance.

Figure 10, showcasing the Relative Error Curve (REC) plot, provides additional insights into the neural network model's adaptability across different levels of error tolerance. This plot further highlights the model's versatility and its capacity to maintain accuracy under varying precision requirements.

In the context of precipitation prediction (Figure 12 and Figure 13), both linear regression and the neural network exhibit high accuracy with low MSE values (0.0068). However, the weak correlation (R-squared value of 0.17) in both models indicates challenges in capturing the underlying patterns in precipitation data. The neural network, while maintaining a weak correlation, showcases an improvement over linear regression.

Figure 13 introduces the REC curve plot for the MLPRegressor model, revealing a Root Mean Squared Error (RMSE) of 0.09. This metric provides a nuanced understanding of the model's overall accuracy in predicting temperature. The lower RMSE suggests that, on average, the predicted temperatures closely align with the actual values.

In conclusion, the neural network model demonstrates superior performance over linear regression in both temperature and precipitation prediction tasks. Its capacity to capture non-linear relationships and adapt to complex data structures makes it a promising choice for meteorological predictions, offering enhanced accuracy and adaptability in diverse forecasting scenarios.

4.2 Challenges in Precipitation Prediction

The relatively low correlation between the test data and predicted data in the context of precipitation prediction can be attributed to several inherent challenges associated with the nature of precipitation patterns. Precipitation is a complex meteorological phenomenon influenced by a myriad of factors, including atmospheric pressure, temperature, wind speed, and humidity, among others. Linear regression, as a simplistic model, may struggle to capture the intricate non-linear relationships among these variables, leading to a limited ability to accurately predict precipitation.

Moreover, precipitation data often exhibit high variability and dependence on localized factors, introducing spatial and temporal complexities. Linear regression models assume a linear relationship between the input features and the target variable, which may oversimplify the intricate dynamics of precipitation patterns. Neural networks, with their capacity to model non-linear relationships and adapt to complex structures, offer an improvement over linear regression in capturing the nuanced interactions that influence precipitation.

In Figure 11 and Figure 12, both linear regression and neural network models exhibit low correlations (R-squared values of approximately 0.17) with the test data. This suggests that the selected features, such as dew point, wind speed, and gust, might not sufficiently encapsulate the multifaceted nature of precipitation. The inherent variability and unpredictability of precipitation events make it challenging for any model, including neural networks, to achieve a high correlation with the observed values.

Additionally, the low correlation may also be indicative of the inherent stochastic nature of precipitation, where short-term variations and local conditions play a

significant role. Linear regression and neural networks may struggle to discern these subtle variations without comprehensive and diverse datasets that encompass a wide range of meteorological conditions.

The challenges associated with the complex, non-linear, and locally influenced nature of precipitation patterns contribute to the relatively low correlations observed in both linear regression and neural network models. Improving precipitation prediction models may require incorporating more sophisticated features, considering localized factors, and exploring advanced techniques that can better capture the intricate dynamics of this meteorological phenomenon.

Conclusion

This project delves into the prediction of daily weather conditions, particularly temperature and precipitation in New York City throughout 2020, utilizing linear regression and neural network models. The dataset, sourced from the National Oceanic and Atmospheric Administration (NOAA), underwent rigorous preprocessing involving data cleaning, feature selection, and visualization. The models were evaluated based on standard metrics such as Mean Squared Error, R-squared values, and Relative Error Curves. Findings indicate that while linear regression exhibited commendable accuracy and correlation in temperature prediction, the neural network model outperformed it, demonstrating enhanced predictive accuracy and improved adaptability across varying error tolerance thresholds. In precipitation prediction, both models achieved high accuracy but struggled with low correlations, attributed to the intricate, non-linear, and locally influenced nature of precipitation patterns.

To enhance the predictive capabilities of weather models, several targeted improvement strategies are proposed. Advanced feature engineering will involve a meticulous selection and transformation of input variables, ensuring a more nuanced representation of meteorological factors influencing temperature and precipitation. Exploring sophisticated neural network architectures entails the investigation of intricate model structures to capture complex non-linear relationships within the data. The implementation of ensemble models, which combine predictions from multiple models, can further enhance accuracy and robustness. Hyperparameter tuning involves fine-tuning model parameters to optimize performance. Consideration of spatial and temporal factors recognizes the localized and time-dependent nature of weather phenomena, introducing refined granularity into the models. Lastly, the utilization of advanced visualization tools aims to provide more insightful and interpretable representations of model outputs, fostering a deeper understanding of the forecasted weather conditions in New York City. Collectively, these targeted enhancements aim to fortify the models' predictive accuracy, ensuring more reliable and precise daily weather forecasts.

references

[1]Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

[2]Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95.

[3]McKinney, W. (2010). Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference (pp. 51-56).

[4]Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.