IntelliHack 5.0

Machine Learning hackathon

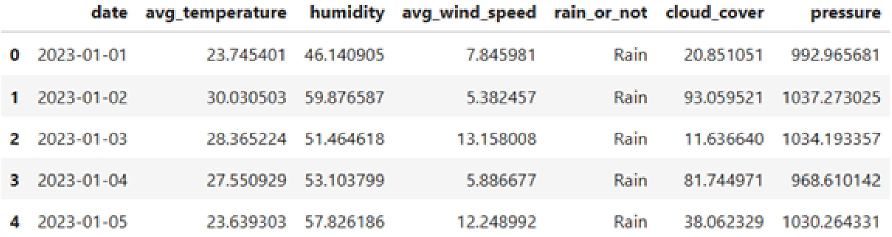
Q1 - Report 01

The Code Rushers

1. Pre-Processing Data and Data Analysis

1.1 Initial Data Inspection

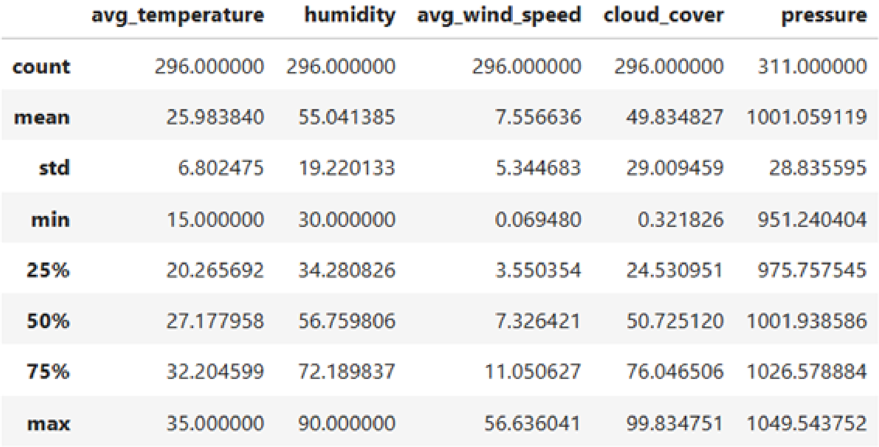
We first checked the structure of the dataset. The dataset contains 5 direct features and one target variable.



	date	avg_temperature	humidity	avg_wind_speed	rain_or_not	cloud_cover	pressure
0	2023-01-01	23.745401	46.140905	7.845981	Rain	20.851051	992.965681
1	2023-01-02	30.030503	59.876587	5.382457	Rain	93.059521	1037.273025
2	2023-01-03	28.365224	51.464618	13.158008	Rain	11.636640	1034.193357
3	2023-01-04	27.550929	53.103799	5.886677	Rain	81.744971	968.610142
4	2023-01-05	23.639303	57.826186	12.248992	Rain	38.062329	1030.264331

Figure 1: Dataset preview

Using `numpy`, we obtained an initial description of the dataset. Through some research, we determined the actual ranges of the data and identified that there are no significant issues with outliers, except for the `avg_wind_speed` feature, which may contain some outliers.

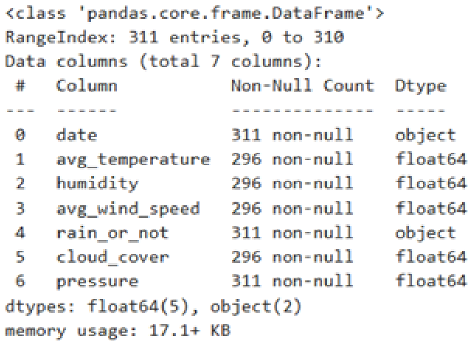


	avg_temperature	humidity	avg_wind_speed	cloud_cover	pressure
count	296.000000	296.000000	296.000000	296.000000	311.000000
mean	25.983840	55.041385	7.556636	49.834827	1001.059119
std	6.802475	19.220133	5.344683	29.009459	28.835595
min	15.000000	30.000000	0.069480	0.321826	951.240404
25%	20.265692	34.280826	3.550354	24.530951	975.757545
50%	27.177958	56.759806	7.326421	50.725120	1001.938586
75%	32.204599	72.189837	11.050627	76.046506	1026.578884
max	35.000000	90.000000	56.636041	99.834751	1049.543752

Figure 2: Dataset Description

1.2 Handling Null Values and Format Errors

Using the `info` function, we identified null values in certain columns. Since these null values were present in only a few columns, we decided to remove them. Additionally, we checked for format errors in numeric columns and found none. For the `rain_or_not` column, we examined the unique values and confirmed that there are only two possible values, indicating no format issues.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 311 entries, 0 to 310
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   date             311 non-null    object
 1   avg_temperature  296 non-null    float64
 2   humidity         296 non-null    float64
 3   avg_wind_speed   296 non-null    float64
 4   rain_or_not      311 non-null    object
 5   cloud_cover      296 non-null    float64
 6   pressure         311 non-null    float64
dtypes: float64(5), object(2)
memory usage: 17.1+ KB
```

Figure 3: Null Values preview

2. EDA Analysis

2.1 Histogram and Box plot Analysis for Features

Histograms were used to visualize the ranges and frequency distributions of the features. Box plots were then used to confirm the presence of outliers in the `avg_wind_speed` feature (indicated by dots). Other features did not exhibit significant outlier issues.

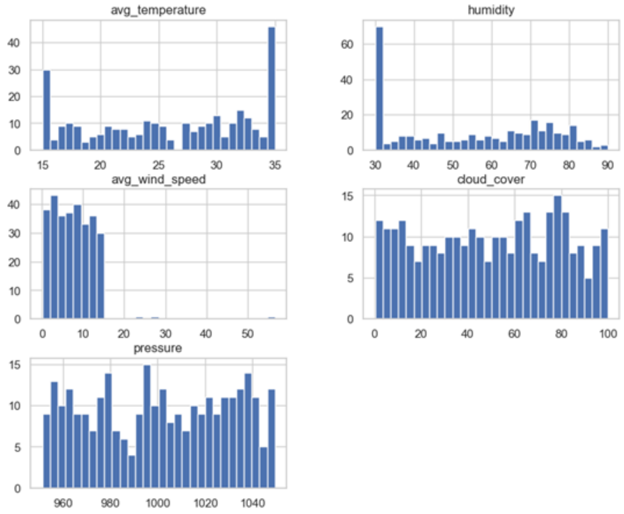


Figure 4: Histogram of Features

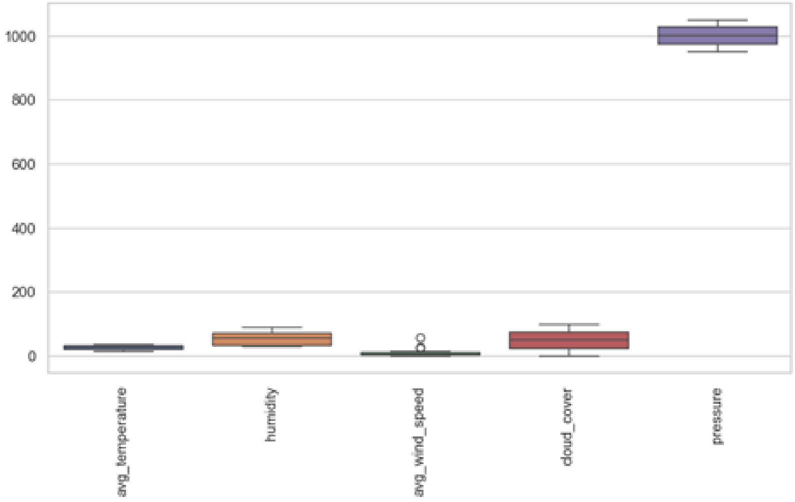


Figure 5: Box Plot of Features

2.2 Correlation Heatmap

A correlation heatmap revealed that most features are nearly independent, except for `humidity` and `avg_temperature`, which have a correlation of 0.9. To reduce redundancy and prevent overfitting, we considered removing one of these features.

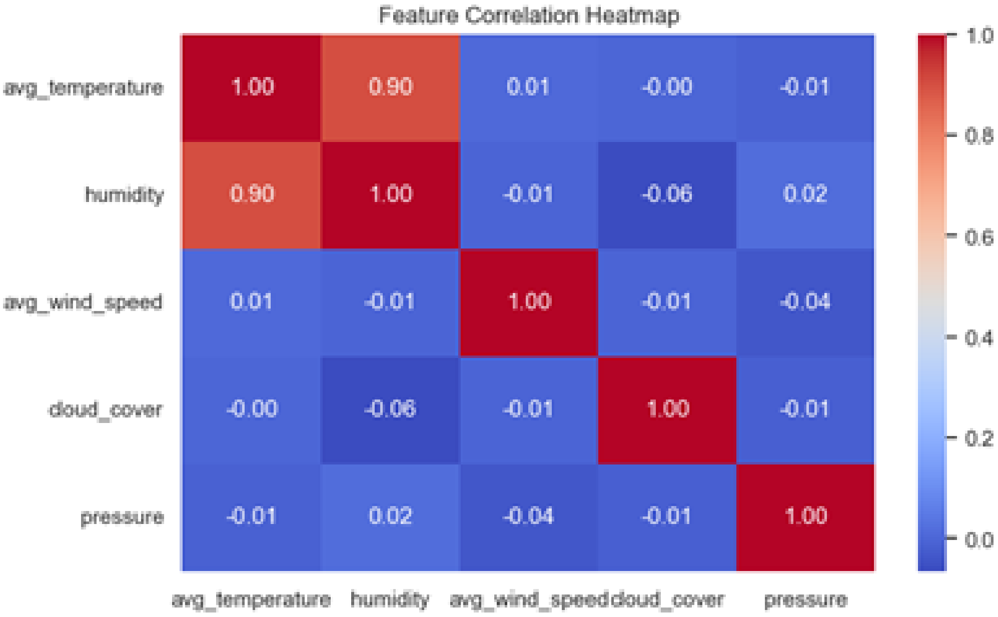


Figure 6: Correlation Heatmap

2.3 Feature-Label Relationship Analysis

We analyzed the relationship between features and the target label to gain insights for feature engineering.

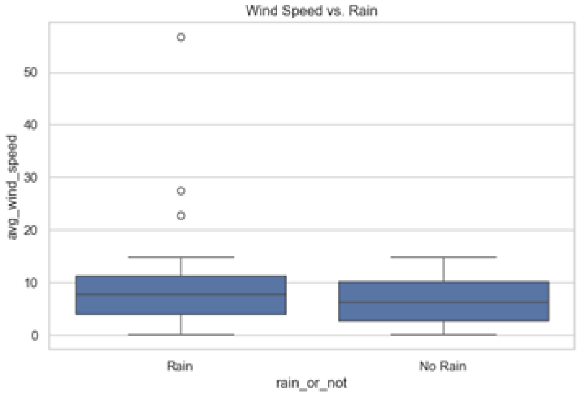


Figure 7

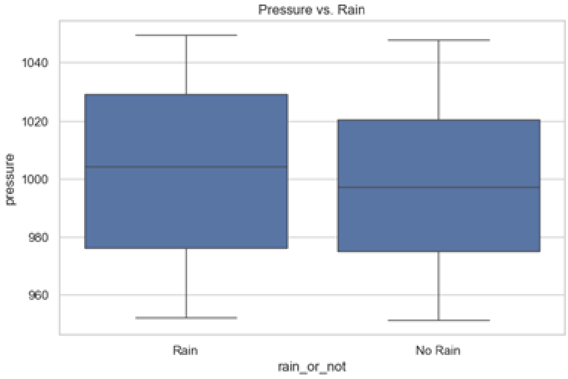


Figure 8

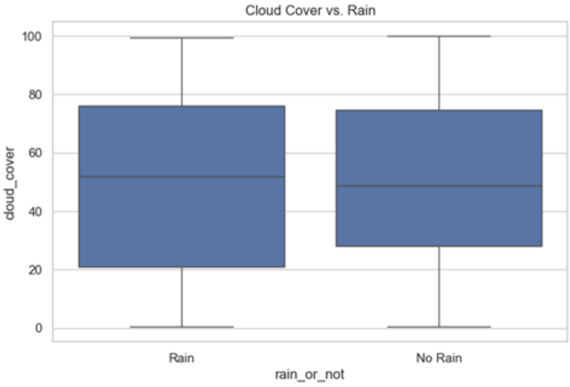


Figure 9

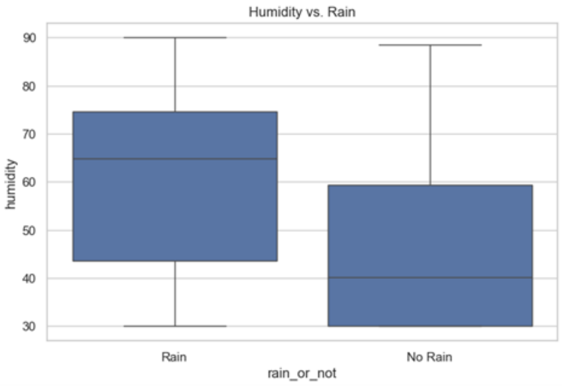


Figure 10

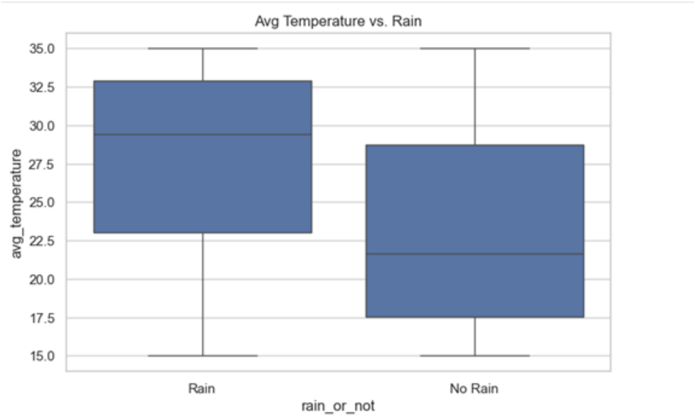
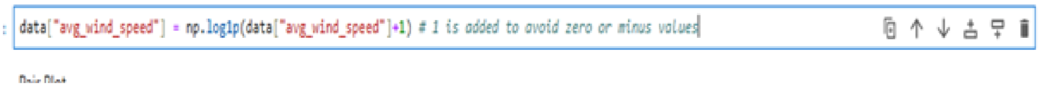


Figure 11

Through this we can again conform that **avg_wind_speed** may have outliers and also we can see that range difference for each features to be rain or Not Rain is small (mostly in same range we can find rain and not rain data points for each features) except **humidty** and temperature. It will make difficult to the model to differentiate label if we are using row features so we have to do feature engineering to create new combined features.

Through some test we have find new features and removed the old features since we only having small dataset having multiple features will reduce accuracy since we have used the data in those features to create new features it's better to remove the old features made to create new ones.

After some test we was able to find that removing the outliers in avg_wind_speed is reducing accuracy as explained in code. so we have done log transform for the feature.



```
data["avg_wind_speed"] = np.log1p(data["avg_wind_speed"]+1) # 1 is added to avoid zero or minus values
```

Pair Plot

Figure 12

2.4 Pair plot

To get more clear one our above decision and easily predict new relation to create new features we can drawn pair plot graph.

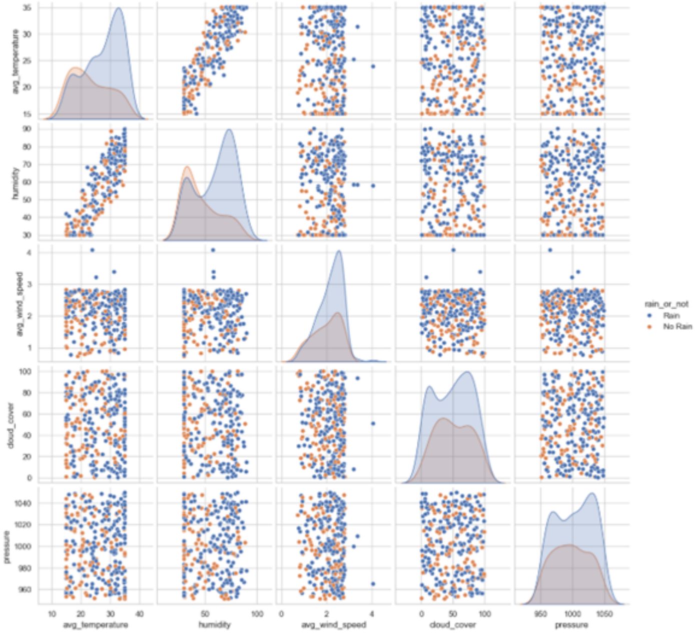
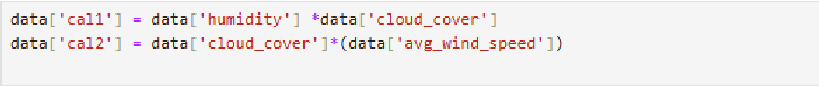


Figure 13: Pair Plot

2.5 Feature Engineering

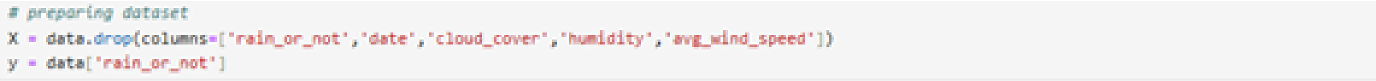
Through testing, we created new features and removed the original ones. Since the dataset is small, having multiple features would reduce accuracy. By using the data from the original features to create new ones, we improved model performance. We also applied a log transformation to the avg_wind_speed feature to handle outliers without reducing accuracy.



```
data['cal1'] = data['humidity'] *data['cloud_cover']
data['cal2'] = data['cloud_cover']*(data['avg_wind_speed'])
```

Figure 14

And removed unwanted features as explained earlier.



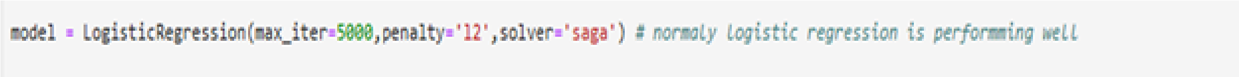
```
# preparing dataset
X = data.drop(columns=['rain_or_not','date','cloud_cover','humidity','avg_wind_speed'])
y = data['rain_or_not']
```

Figure 15

3. Model Training and Evaluation

3.1 Model Selection

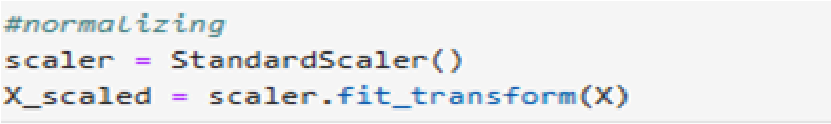
We tested several models, including logistic regression, decision trees, random forests, and gradient boosting. Logistic regression performed best on the test dataset, achieving 75% accuracy with 300 data points.



```
model = LogisticRegression(max_iter=5000,penalty='l2',solver='saga') # normaly logistic regression is performming well
```

Figure 16

We tuned the hyperparameters to optimize performance and normalized the feature values to account for their varying ranges.

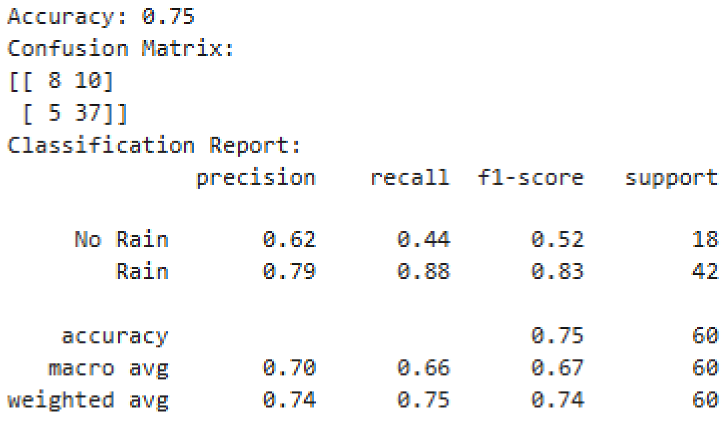


```
#normalizing
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Figure 17

3.2 Final Single-Day Prediction Model

The final model was used to predict single-day rain probabilities. The results are shown below:



```
Accuracy: 0.75
Confusion Matrix:
[[ 8 10]
 [ 5 37]]
Classification Report:
              precision    recall  f1-score   support

     No Rain       0.62      0.44      0.52        18
        Rain       0.79      0.88      0.83        42

    accuracy                           0.75        60
   macro avg       0.70      0.66      0.67        60
weighted avg       0.74      0.75      0.74        60
```

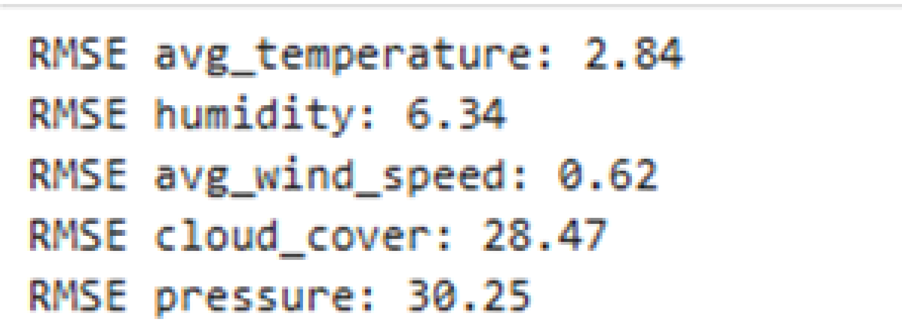
Figure 18: Final Ouput

4. 21-Day Prediction Model

We built a model to predict rain probabilities for the next 21 days using `XGBRegressor`. Lagging features were created in the dataset to predict future values of the five original features. These predictions were then fed into the single-day prediction model to generate 21-day forecasts. The prediction accuracy was evaluated using RMSE.

Error Analysis

The RMSE for each feature's 21-day predictions is shown below. The error percentage is relatively small compared to the feature values, except for `cloud_cover`. Increasing the dataset size could further reduce this error.

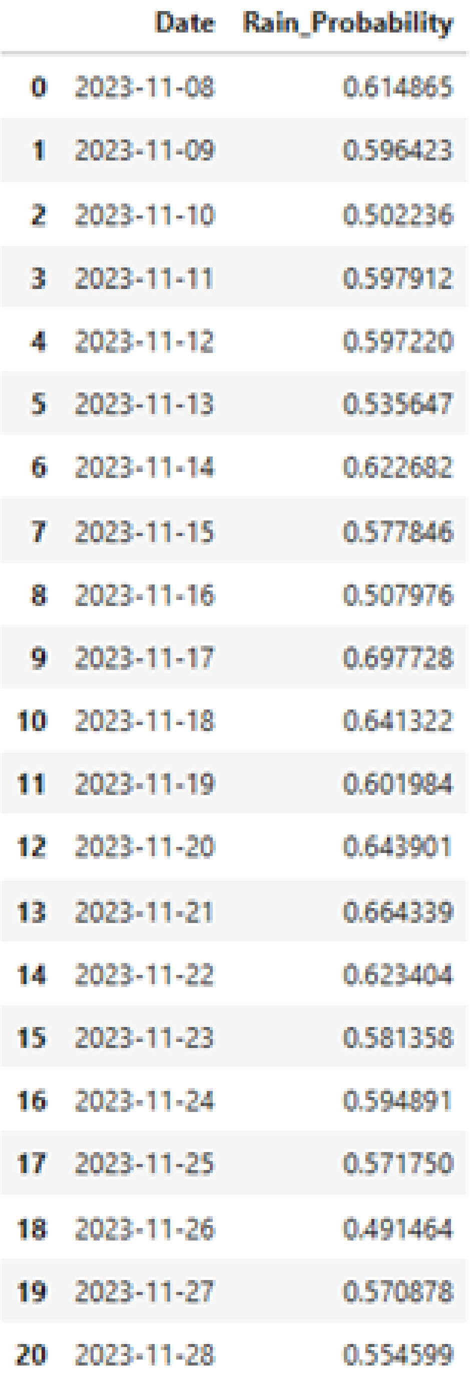


```
RMSE avg_temperature: 2.84
RMSE humidity: 6.34
RMSE avg_wind_speed: 0.62
RMSE cloud_cover: 28.47
RMSE pressure: 30.25
```

Figure 19: 21-Day Prediction Results

5. Final Results

The final results demonstrate the effectiveness of the model in predicting rain probabilities for both single-day and 21-day forecasts. Further improvements can be achieved by incorporating additional data and refining feature engineering techniques.



	Date	Rain_Probability
0	2023-11-08	0.614865
1	2023-11-09	0.596423
2	2023-11-10	0.502236
3	2023-11-11	0.597912
4	2023-11-12	0.597220
5	2023-11-13	0.535647
6	2023-11-14	0.622682
7	2023-11-15	0.577846
8	2023-11-16	0.507976
9	2023-11-17	0.697728
10	2023-11-18	0.641322
11	2023-11-19	0.601984
12	2023-11-20	0.643901
13	2023-11-21	0.664339
14	2023-11-22	0.623404
15	2023-11-23	0.581358
16	2023-11-24	0.594891
17	2023-11-25	0.571750
18	2023-11-26	0.491464
19	2023-11-27	0.570878
20	2023-11-28	0.554599

Figure 20: Final 21-Days Results