Project Presentation

DSA210

Kiyan Salehniya

Weather-Driven Sales Prediction in NYC Coffee Shops

Motivation

I always was curious if the weather conditions does affect people tendencies in the coffee shop or not. For example, we always hear from our friends that “I love drinking coffee in rainy or cloudy weather” or “I love drinking hot chocolate when it is snowing”. However, is it happening in real life and can coffee shop brands predict their sales in different weather conditions? Understanding how **weather conditions affect coffee shop sales** can help coffee shops optimize their inventory, staffing, and marketing strategies. This project investigates how weather impacts product category preferences (coffee, tea, bakery, etc.) using real sales and weather data.

Data Sources

- **Coffee Sales Data**: Kaggle (3 coffee shop brands from NYC) Link: Kaggle Dataset
- **Weather Data**: VisualCrossing.com (hourly weather for NYC, Jan–Jun 2023)

Data Analysis & Preprocessing

1. Data Integration & Matching:

- Matched hourly weather with minute-based transaction data based on the hour of transaction.

2. Feature Engineering:

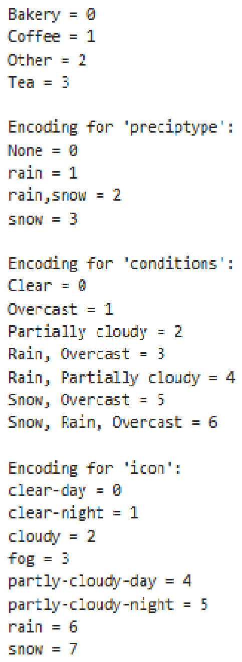
- Created features like humidity_prec_percentage and cloud_rain_percentage
- Extracted hour and weekday from timestamps

3. Handling Missing Values:

- Filled preciptype NaNs with "None"

4. Encoding & Simplification:

- Encoded categorical features (weather, product categories, etc.)
- Merged rare product categories into a new "Other" class (now 4 total: Coffee, Tea, Bakery, Other)



```
Bakery = 0
Coffee = 1
Other = 2
Tea = 3

Encoding for 'preciptype':
None = 0
rain = 1
rain,snow = 2
snow = 3

Encoding for 'conditions':
Clear = 0
Overcast = 1
Partially cloudy = 2
Rain, Overcast = 3
Rain, Partially cloudy = 4
Snow, Overcast = 5
Snow, Rain, Overcast = 6

Encoding for 'icon':
clear-day = 0
clear-night = 1
cloudy = 2
fog = 3
partly-cloudy-day = 4
partly-cloudy-night = 5
rain = 6
snow = 7
```

5. Standardization & Splitting:

- Scaled numeric data for consistency
- 80-20 train-test split

Hypothesis Testing

Goal: Determine if weather affects product category choice.

Test Used: One-Way ANOVA

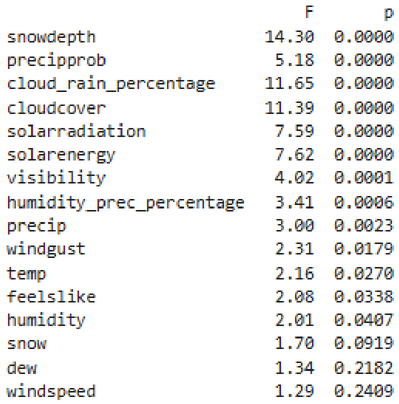
Null Hypothesis (H0): Weather does **not** affect sales category
Alternative (H1): Weather **does** affect sales category

Significant Weather Features ($p < 0.05$):

- snowdepth, cloudcover, precipprob, solarradiation, temp, feelslike, visibility, etc.

Non-significant Features:

- snow, dew, windspeed



```
                                F       p
snowdepth                   14.30  0.0000
precipprob                   5.18  0.0000
cloud_rain_percentage       11.65  0.0000
cloudcover                  11.39  0.0000
solarradiation               7.59  0.0000
solarenergy                  7.62  0.0000
visibility                   4.02  0.0001
humidity_prec_percentage     3.41  0.0006
precip                       3.00  0.0023
windgust                     2.31  0.0179
temp                         2.16  0.0270
feelslike                    2.08  0.0338
humidity                     2.01  0.0407
snow                         1.70  0.0919
dew                          1.34  0.2182
windspeed                    1.29  0.2409
```

Machine Learning Pipeline

1. Models Evaluated:

- Decision Tree
- Random Forest
- KNN
- XGBoost
- Naive Bayes
- Gradient Boosting
- Logistic Regression

2. Evaluation Criteria:

- Accuracy
- Fairness (recall for all classes)
- Confusion Matrix and Classification Report

3. Imbalance Handling:

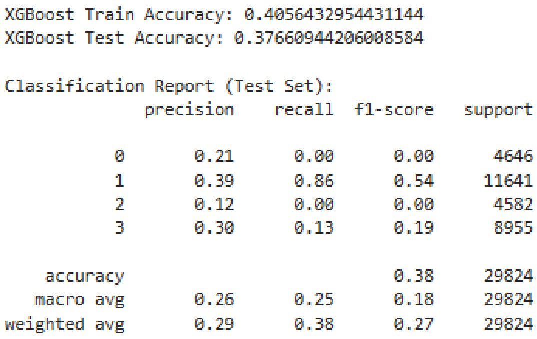
- Used **SMOTE** on KNN and Logistic Regression to improve class balance or use sample weights

4. Hyperparameter Tuning:

- GridSearchCV for XGBoost and Random Forest
- Tried with and without sample weights

Findings

- **XGBoost Base Model** had the best trade-off between accuracy and fairness.



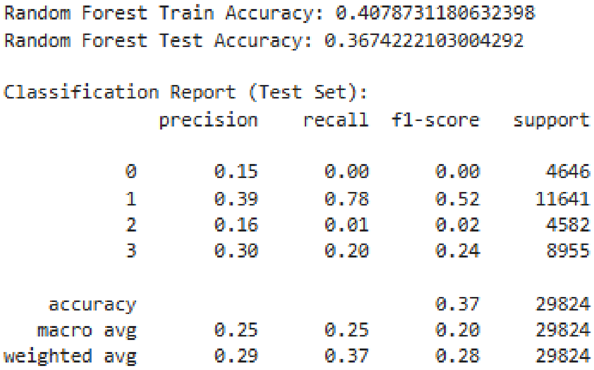
```
XGBoost Train Accuracy: 0.4056432954431144
XGBoost Test Accuracy: 0.37660944206008584

Classification Report (Test Set):
              precision    recall  f1-score   support

           0       0.21      0.00      0.00      4646
           1       0.39      0.86      0.54     11641
           2       0.12      0.00      0.00      4582
           3       0.30      0.13      0.19      8955

    accuracy                           0.38     29824
   macro avg       0.26      0.25      0.18     29824
weighted avg       0.29      0.38      0.27     29824
```

- **Random Forest Base Model** was a close second.



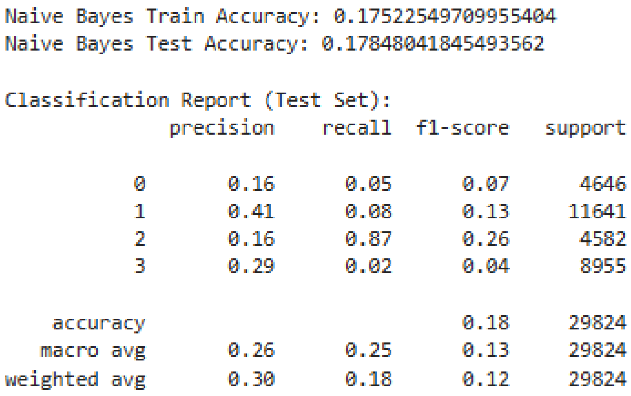
```
Random Forest Train Accuracy: 0.4078731180632398
Random Forest Test Accuracy: 0.3674222103004292

Classification Report (Test Set):
              precision    recall  f1-score   support

           0       0.15      0.00      0.00      4646
           1       0.39      0.78      0.52     11641
           2       0.16      0.01      0.02      4582
           3       0.30      0.20      0.24      8955

    accuracy                           0.37     29824
   macro avg       0.25      0.25      0.20     29824
weighted avg       0.29      0.37      0.28     29824
```

- **Sample weights** improved class balance and prediction fairness but dropped accuracy.
- **Naive Bayes** performed the worst.



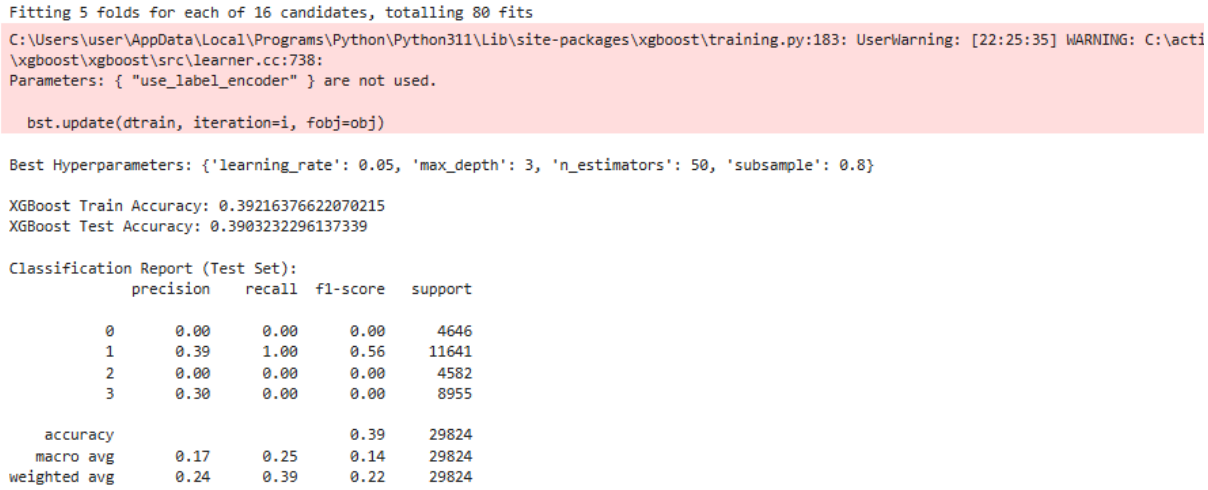
```
Naive Bayes Train Accuracy: 0.17522549709955404
Naive Bayes Test Accuracy: 0.17848041845493562

Classification Report (Test Set):
              precision    recall  f1-score   support

           0       0.16      0.05      0.07      4646
           1       0.41      0.08      0.13     11641
           2       0.16      0.87      0.26      4582
           3       0.29      0.02      0.04      8955

    accuracy                           0.18     29824
   macro avg       0.26      0.25      0.13     29824
weighted avg       0.30      0.18      0.12     29824
```

- **Grid Search models** often overfit the dominant class.



```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
C:\Users\user\AppData\Local\Programs\Python\Python311\Lib\site-packages\xgboost\training.py:183: UserWarning: [22:25:35] WARNING: C:\acti
\xgboost\xgboost\src\learner.cc:738:
Parameters: { "use_label_encoder" } are not used.

  bst.update(dtrain, iteration=i, fobj=obj)

Best Hyperparameters: {'learning_rate': 0.05, 'max_depth': 3, 'n_estimators': 50, 'subsample': 0.8}

XGBoost Train Accuracy: 0.39216376622070215
XGBoost Test Accuracy: 0.3903232296137339

Classification Report (Test Set):
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      4646
           1       0.39      1.00      0.56     11641
           2       0.00      0.00      0.00      4582
           3       0.30      0.00      0.00      8955

    accuracy                           0.39     29824
   macro avg       0.17      0.25      0.14     29824
weighted avg       0.24      0.39      0.22     29824
```

Limitations

- Some class imbalance still persists, and I need more data for the imbalance data and make data more balance

- Predictions rely heavily on accurate timestamp-weather matching
- My sales data is limited to 6 months only
- Imbalanced data decrease training accuracy so much

Future Work

- Include full-year weather data and sales data
- Incorporate more product metadata (e.g., promotions, ingredients)
- Try deep learning or time series models
- Build a dashboard for real-time prediction support

Summary

By combining sales and weather data, we could successfully identified key weather patterns that influence product category sales, but with really low accuracy. XGBoost proved to be the most effective model, providing both accuracy and general fairness across categories. These insights can help a little coffee shop managers anticipate demand and make smarter, weather-aware business decisions.