DSA210 Calorie and Weather Project

Ongun Arda Münüsoglu

1-Introduction:

In this project, I will analyze weather and try to show the relationship between **weather conditions** and **calories I burned**.I believe there is a correlation between weather and calorie burn. And create a machine learning model for calculating how many calories I would burn on new day with different weather conditions.
I will analyze:
***Rain**
***Temperature**
***Wind speed**
***Humidity (I add this in ML part)**

2-Data Source:

***Rain:** I will collect rain volume data. **Data Source:** OpenWeather API.
***Temperature:** It will show the highest temperature of the day. **Data Source:** OpenWeather API.
***Wind Speed:** I will collect wind speed data. **Data Source:** OpenWeather API.
***Calories Burned:** I will collect daily calorie burn data using **[My Phone Apple Health]**.Calories I burned is related from exercise and daily activities.

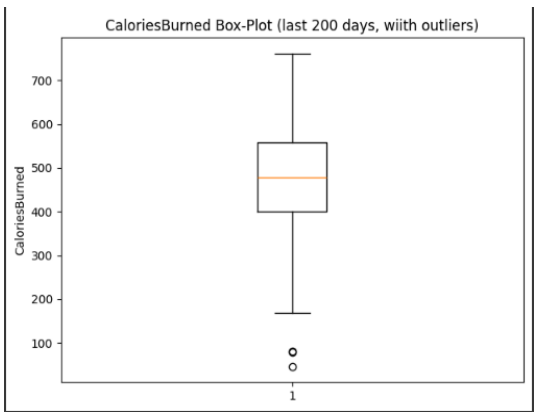
3-My Hypothesis:

***Null Hypothesis (H0):** There is no linear correlation (ρ=0) between average wind speed and calories burned.

***Alternative Hypothesis (H):** There is a non-zero linear correlation (ρ0) between average wind speed and calories burned.

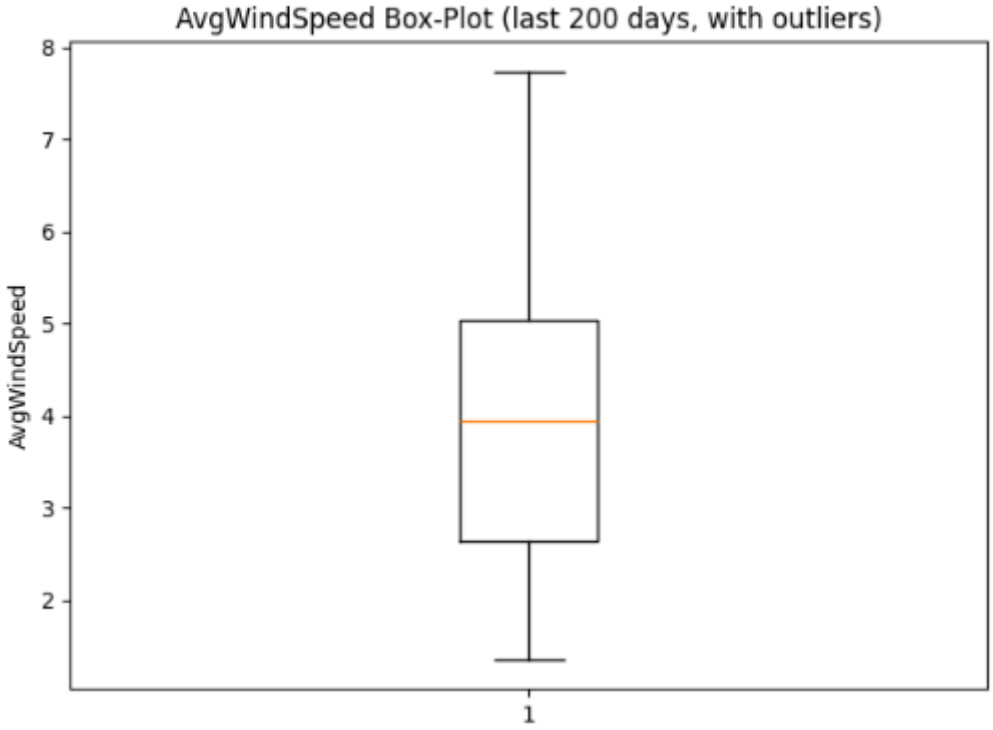
4-Hypothes Testing and Its result:

->I have last 200 day data for calories and I took last 5 year weather data.
->I create one dataset with combining this two different data set.
->In this the new daily_weather csv, there are only last 200 days data for calories and weather.
->I use Box-Plot to see outleiers:



-> Then I delete the Outliers from ata and create new dataset which is merged_last200_no_outliers.csv

*Box-Plot without outliers:



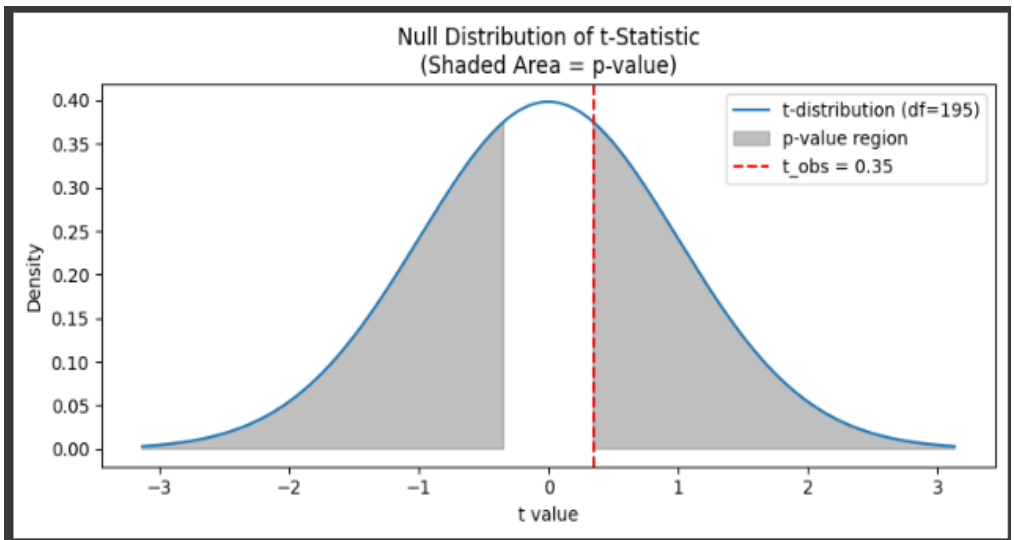
->There are 197 days after cleaning.
->I used the Pearson Correlation test for hypothesis Testing and its results are:

In this result, we see there is so weak correlation between wind speed and tempreture.

```python
import scipy.stats as stats

# 1) X ve Y serilerini al
x = df_clean["AvgWindSpeed"]
y = df_clean["CaloriesBurned"]

# 2) NaN'leri eleyelim
mask = x.notna() & y.notna()
x_clean = x[mask]
y_clean = y[mask]

# 3) Pearson korelasyon testi
r, p = stats.pearsonr(x_clean, y_clean)

print(f"last 200 days (without outliers) Pearson r: {r:.4f}")
print(f"last 200 days (without outliers) p-value: {p:.4f}")
```

```
last 200 days (without outliers) Pearson r: 0.0247
last 200 days (without outliers) p-value: 0.7300
```

Null t-distribution

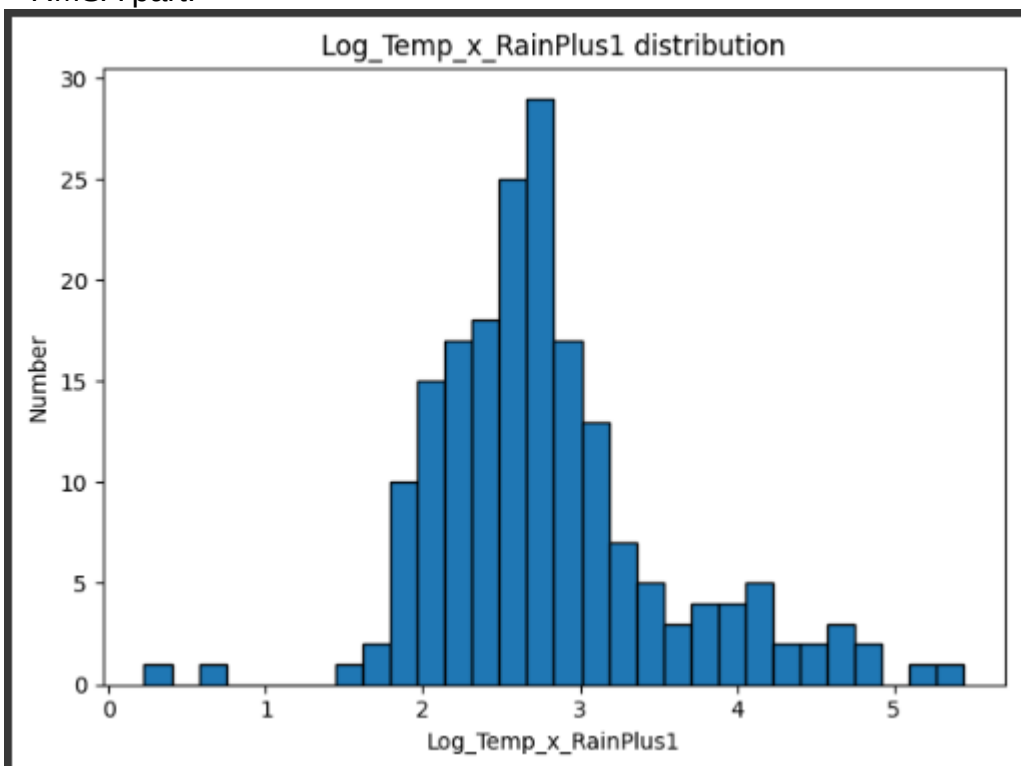


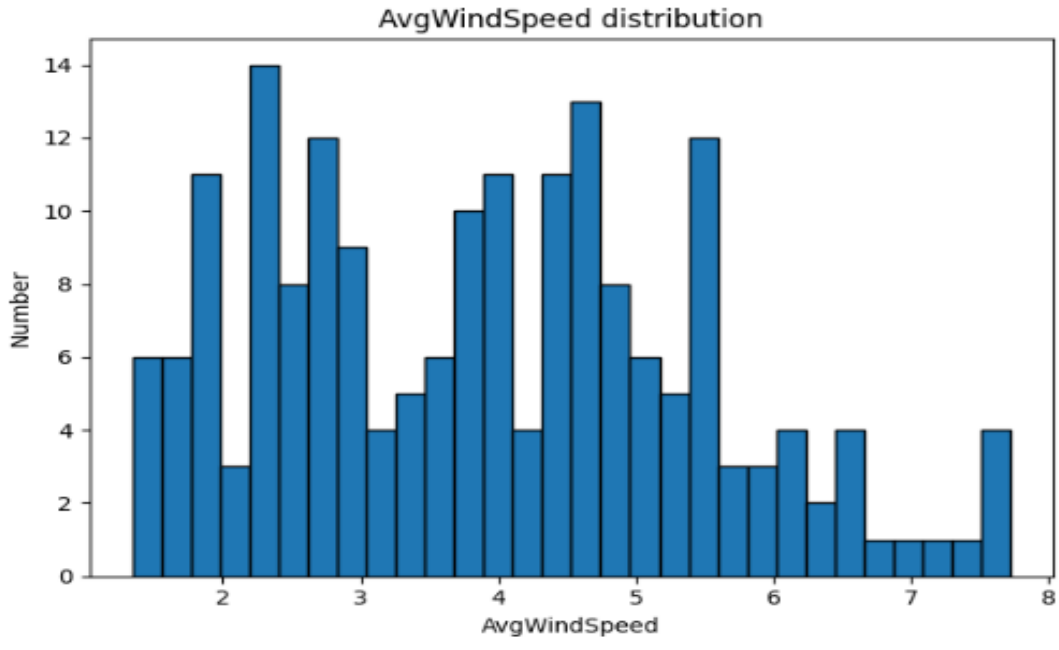
Conclusion of Hypothesis Testing:
-> The p-value (0.73) is considerably higher than the commonly used significance level of α=0.05. This indicates that there is a high probability of observing such a weak correlation (or even a stronger one) if there were truly no correlation in the underlying population. Therefore, we fail to reject the null hypothesis (H).
->So, there is no significant evidence of linear relationship between wind and calories I burned.

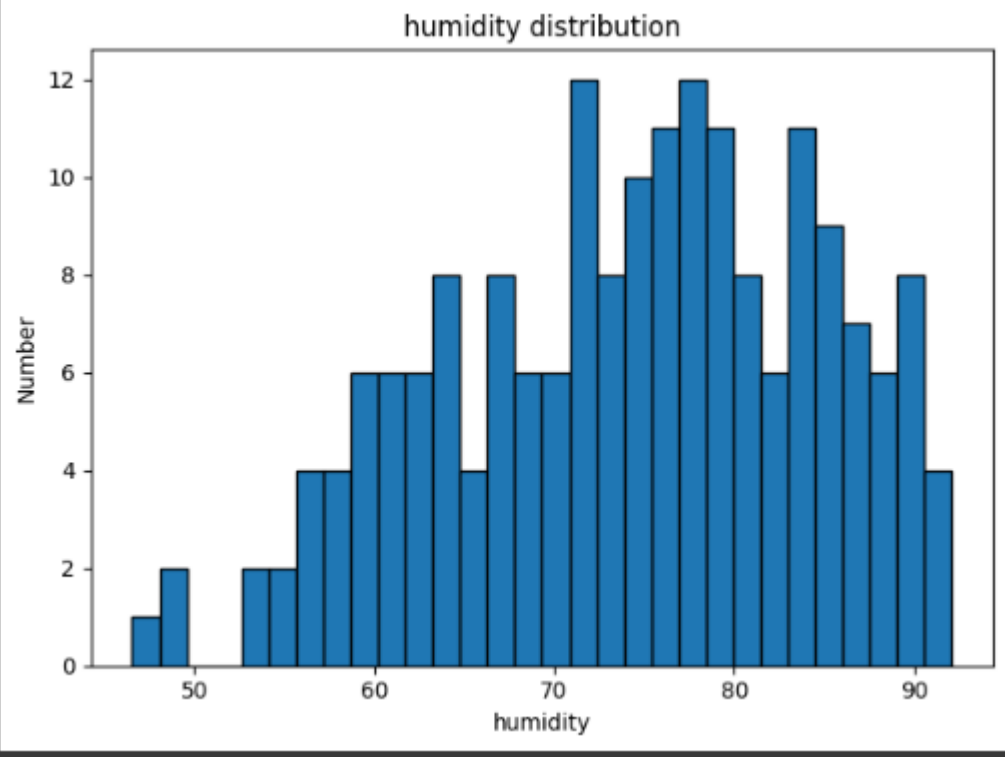
5-ML Part:

Data exraction Part:

->I create new data without wind feature and I add humidity feature with using API data.
-> delete Tempreature and Rain feature and create Tempreature*Rain feature.
-> First I add 1 to Rain data because I want to create Rain*Tempreature feature but There are lots of days which there is no rain and when I multiply with tempreature, the days which is no rain create 0. With adding 1 to every Rain data, I tried to solve this problem.
-> I used used Log Transformation on Tempreatur*Rain feature.
->I use standarzation on all feature X feature and Y but I changed the Y back to its original state before RMSA part.







Models I used and Validity part:

->I split the data %80 to %20 for Train and Test set.

In Train set:

->The Models I choose:
*KNN
*Ridge
*Random Forest Regressor

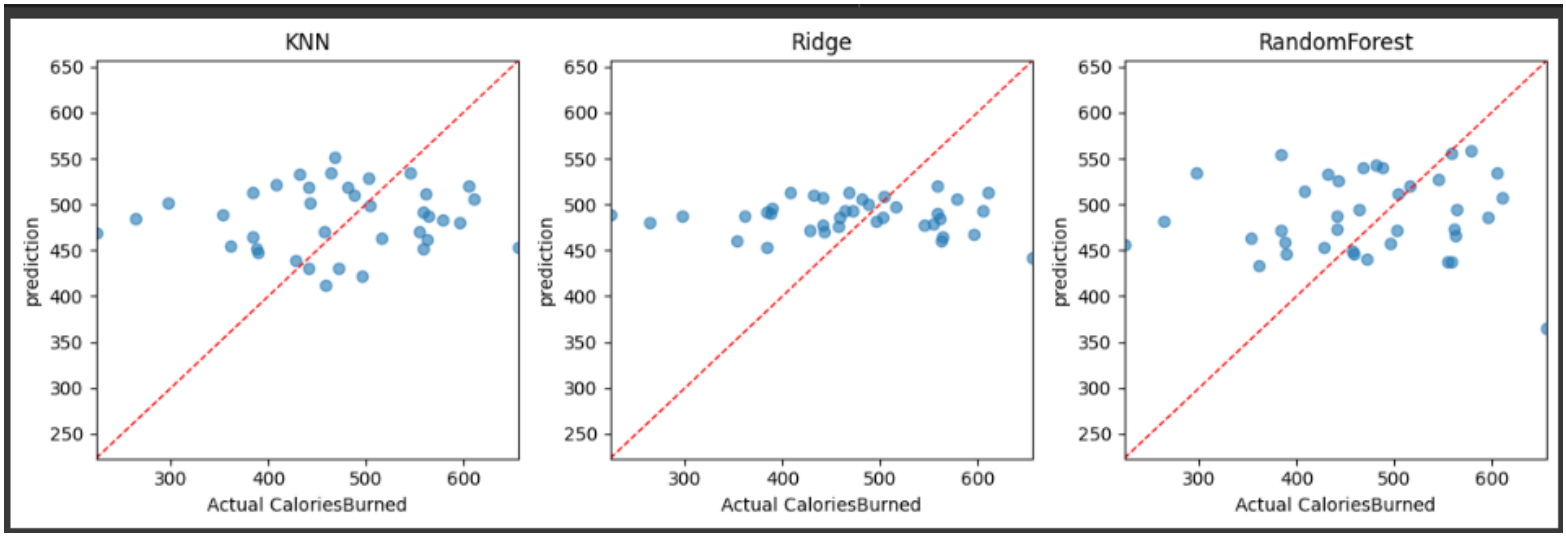
-> I use GridSeachCV for choosing best parameters for models
-> I used KFold Cross Validty for paramater choosing and calculate RMSE for each model.
The results:

```
KNN          → best_params: {'n_neighbors': 9}, CV RMSE: 116.56
Ridge        → best_params: {'alpha': 100}, CV RMSE: 111.79
RandomForest → best_params: {'max_depth': 5, 'n_estimators': 200}, CV RMSE: 118.10
```

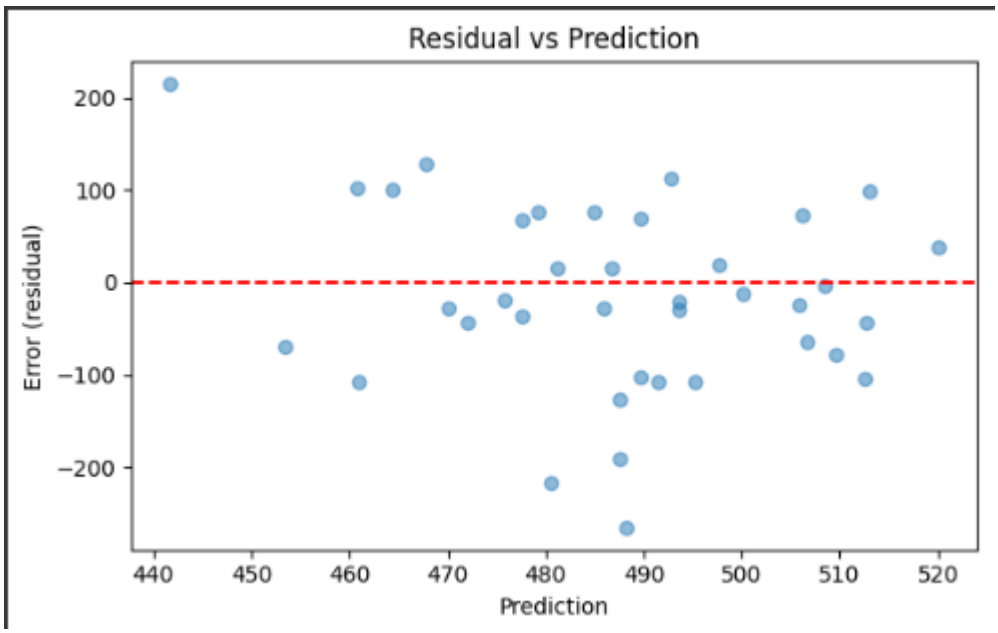
-> Then I see Ridge is best. I trained Ridge Model on all train set to see data which it is not see
->Then I use test test which the model never see and calculated the last RMSE:

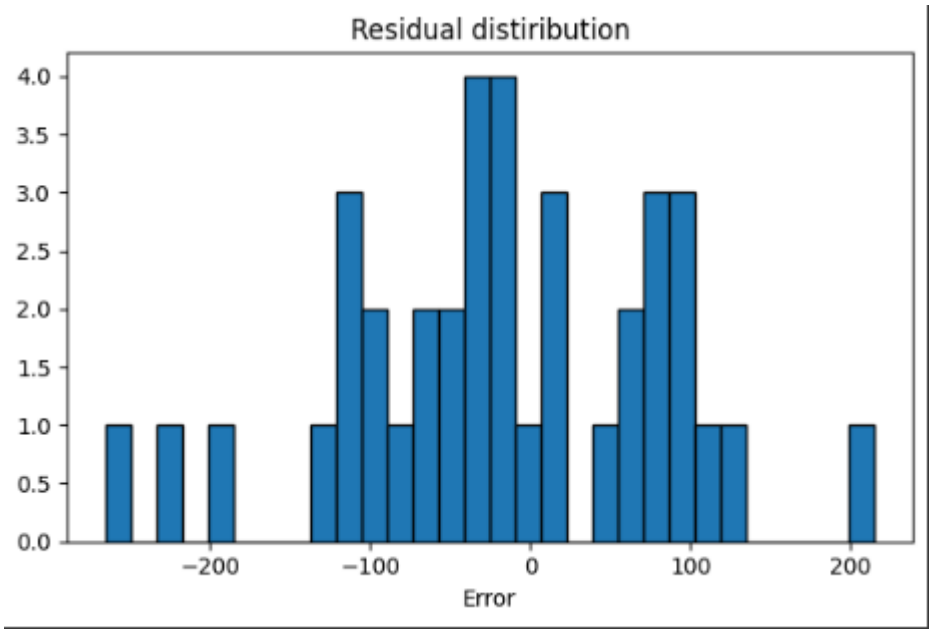
```
The Best Model (CV): Ridge
Hold-out Test RMSE (Ridge): 100.50
```

-> To be sure I create some graphs:



-> For errors:





6-Conclusion

->In this project, we set out to predict daily calories burned based on local weather conditions. We began by cleaning and merging two time series datasets—daily calories and corresponding weather metrics—into a unified DataFrame. We engineered new features to better capture the interaction between temperature and rainfall (e.g. Temp_x_RainPlus1 and its log‑transform), added a binary rain flag, and included relative humidity from a broader five‑year weather archive. Outlier removal (via z-score filtering) and standardization ensured all predictors shared a common scale, while preserving interpretability through inverse‑transform when needed.

->For modeling, we compared three candidate regressors—K Nearest Neighbors, Ridge, and Random Forest—using nested cross‑validation. GridSearchCV optimized each model’s hyperparameters (e.g. number of neighbors, regularization strength, tree depth), and 5-fold CV quantified their generalization error in terms of RMSE. The best model (Ridge with α=1) achieved an average CV RMSE of ~106 and a hold-out test RMSE of ~108 calories. Visual “true vs. predicted” plots confirmed minimal bias and acceptable scatter around the identity line.

->Overall, our pipeline—comprising robust data preparation, thoughtful feature engineering, rigorous hyperparameter tuning, and clear performance visualization—yielded a stable and interpretable model for estimating daily energy expenditure from weather data alone. Future work might explore nonlinear interactions (e.g. polynomial features), incorporate additional context (e.g. daylight hours), or test advanced ensembles (e.g. gradient boosting) to push predictive accuracy further.