# Exploring Tools for Interpretable Machine Learning

Dr. Juan Orduz

PyData Global 2021

PyData Global

# Outline



PyData Global

# Introduction

## Aim and Scope of the Talk

**What?** In this talk we want to test various ways of getting a better understanding on how machine learning (ML) models generate predictions and how features interact with each other. This is in general not straight forward and key components are

- Domain knowledge on the problem.
- Understanding on the input data.
- Understanding the logic behind the ML algorithms.

**How?** We are going to wor out a concrete example.

# Introduction

## Aim and Scope of the Talk

**What?** In this talk we want to test various ways of getting a better understanding on how machine learning (ML) models generate predictions and how features interact with each other. This is in general not straight forward and key components are

- Domain knowledge on the problem.
- Understanding on the input data.
- Understanding the logic behind the ML algorithms.

**How?** We are going to wor out a concrete example.

## References

This talk is based on this blog post ([3]), which itself is based on these two amazing references:

- Interpretable Machine Learning, A Guide for Making Black Box Models Explainable by Christoph Molnar
- Interpretable Machine Learning with Python by Serg Masís

# Interpretable ML $\neq$ Causality

Note that the methods discussed in this notebook are not related with causality.

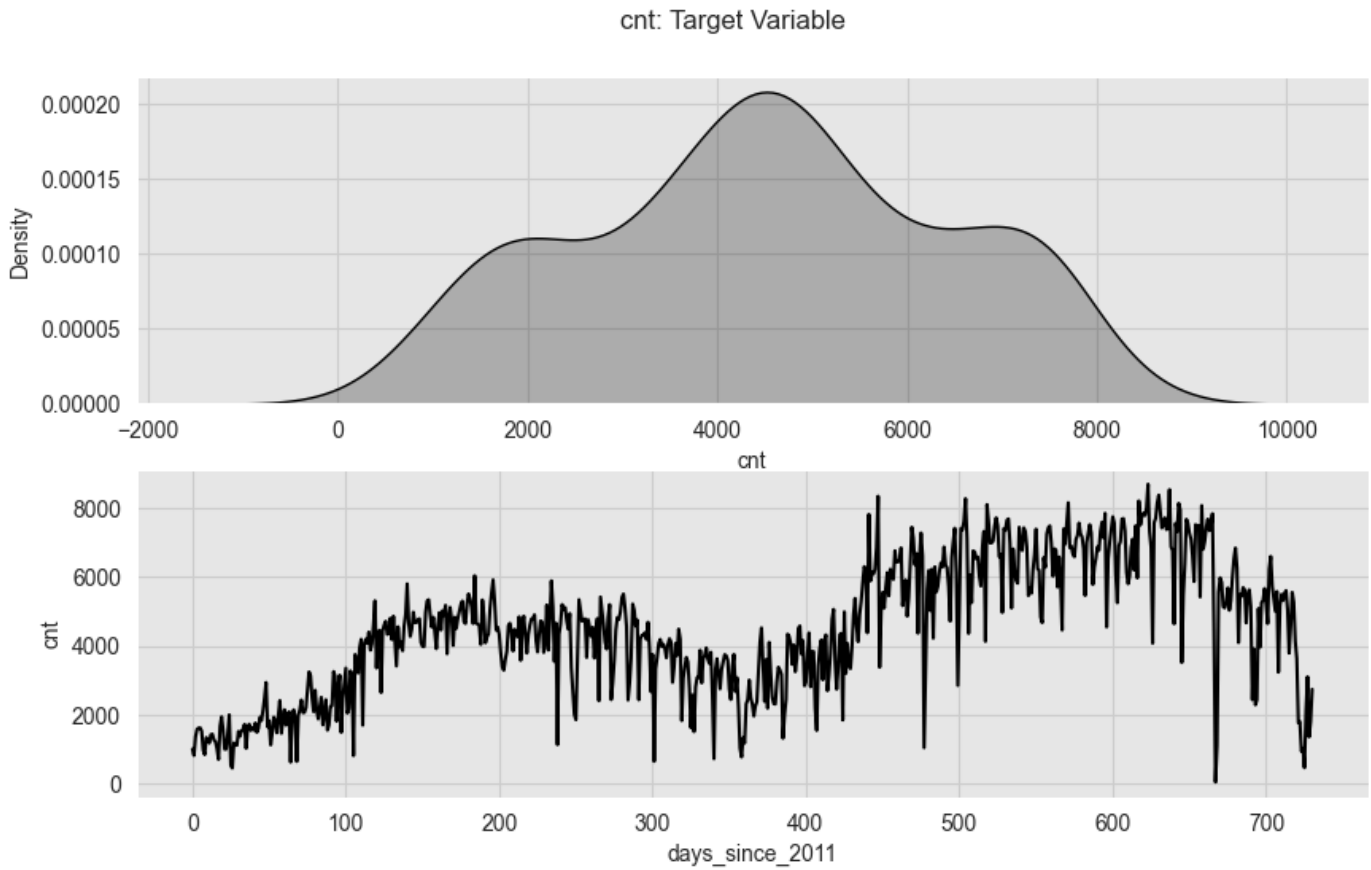
## References

- Be Careful When Interpreting Predictive Models in Search of Causal Insights by Scott Lundberg(one of the core developers of SHAP)
- Statistical Rethinking, A Bayesian Course with Examples in R and Stan by Richard McElreath
- Causal Inference: The Mixtape by Scott Cunningham

PyData Global

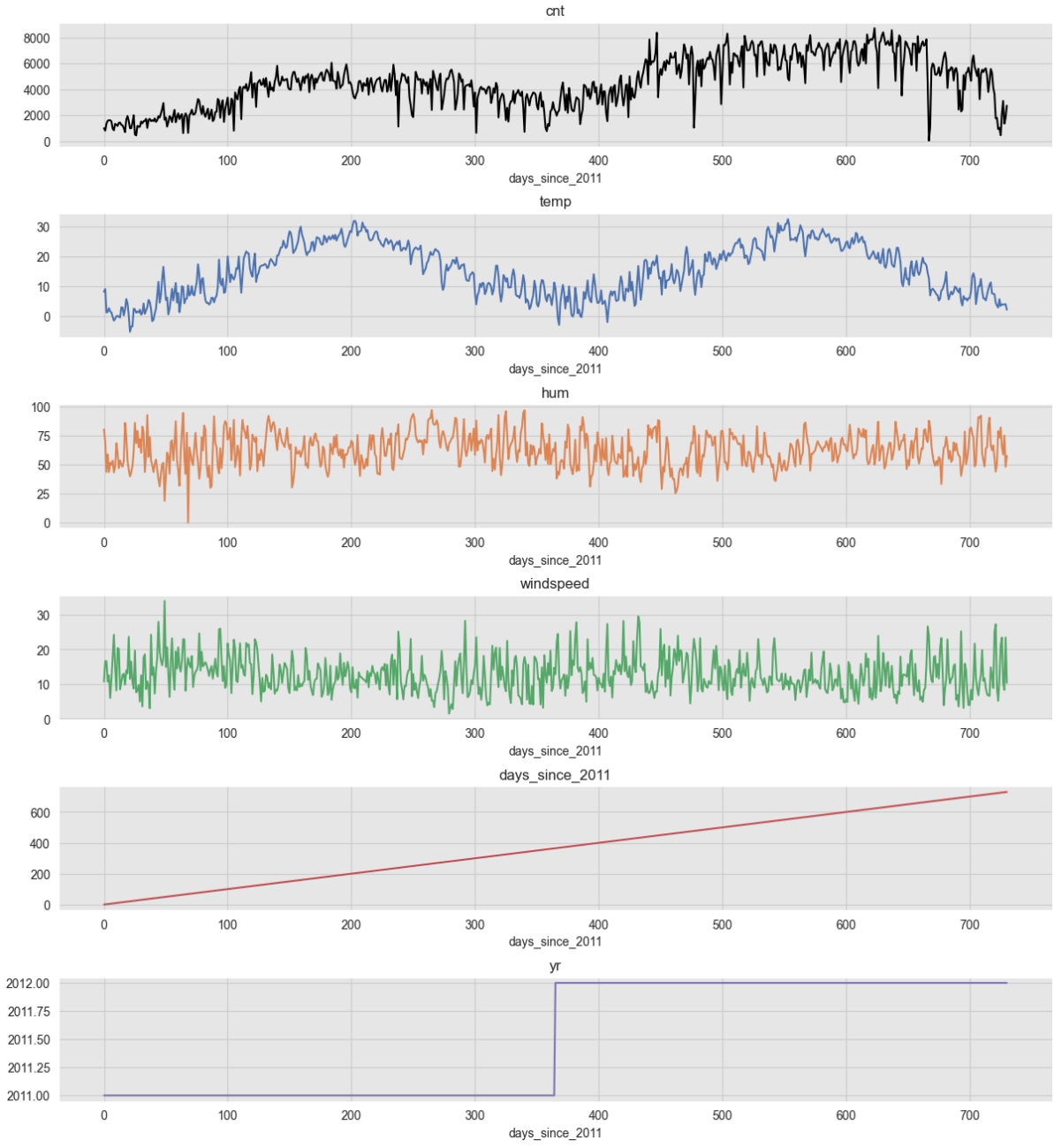
# Target Variable - cnt: Daily Bike Rents

| | season | yr | mnth | holiday | weekday | workingday | weathersit | temp | hum | windspeed | cnt | days_since_2011 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | SPRING | 2011 | JAN | NO HOLIDAY | SAT | NO WORKING DAY | MISTY | 8.175849 | 80.5833 | 10.749882 | 985 | 0 |
| 1 | SPRING | 2011 | JAN | NO HOLIDAY | SUN | NO WORKING DAY | MISTY | 9.083466 | 69.6087 | 16.652113 | 801 | 1 |
| 2 | SPRING | 2011 | JAN | NO HOLIDAY | MON | WORKING DAY | GOOD | 1.229108 | 43.7273 | 16.636703 | 1349 | 2 |
| 3 | SPRING | 2011 | JAN | NO HOLIDAY | TUE | WORKING DAY | GOOD | 1.400000 | 59.0435 | 10.739832 | 1562 | 3 |
| 4 | SPRING | 2011 | JAN | NO HOLIDAY | WED | WORKING DAY | GOOD | 2.666979 | 43.6957 | 12.522300 | 1600 | 4 |



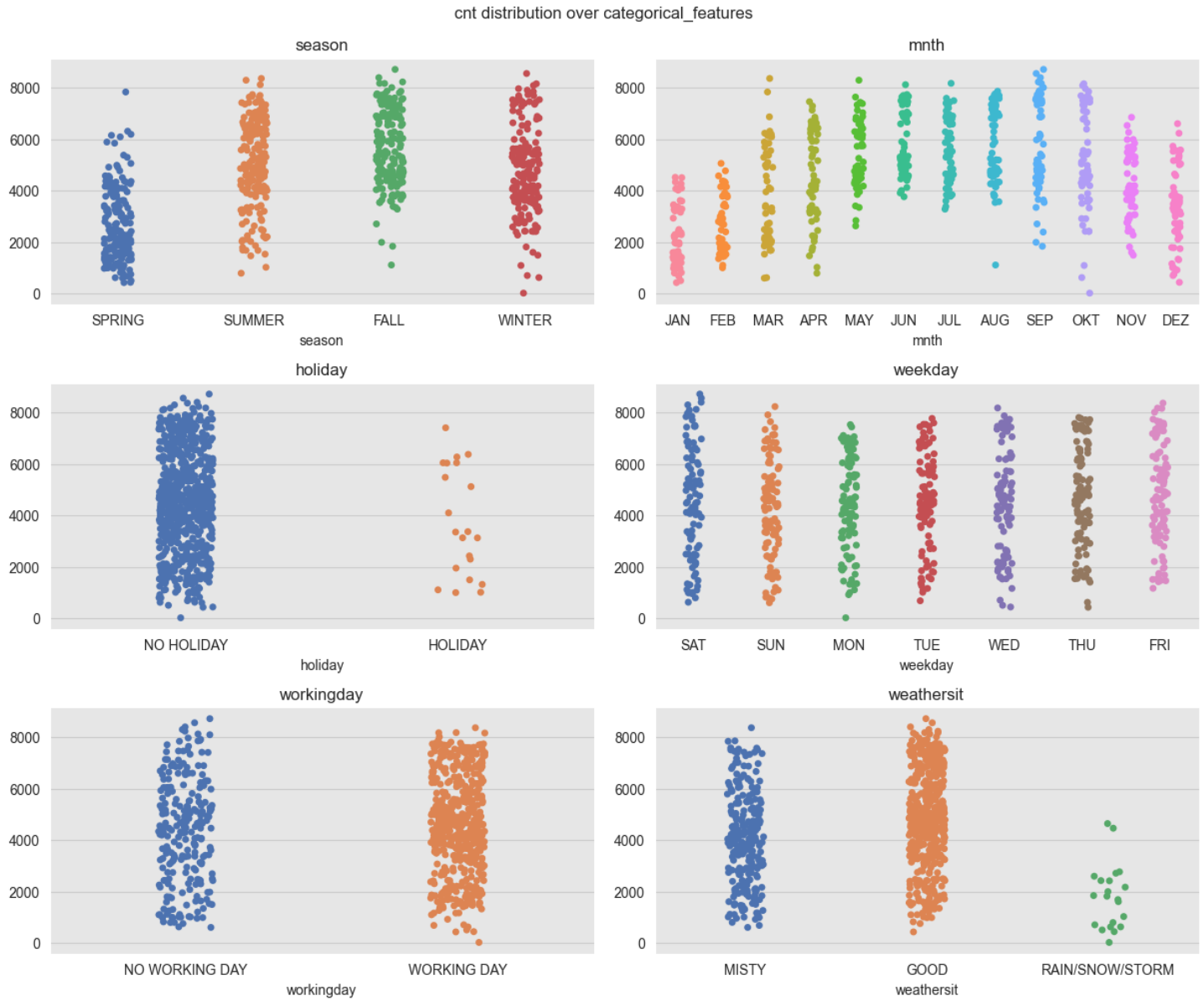

PyData Global

# Continuous Regressors



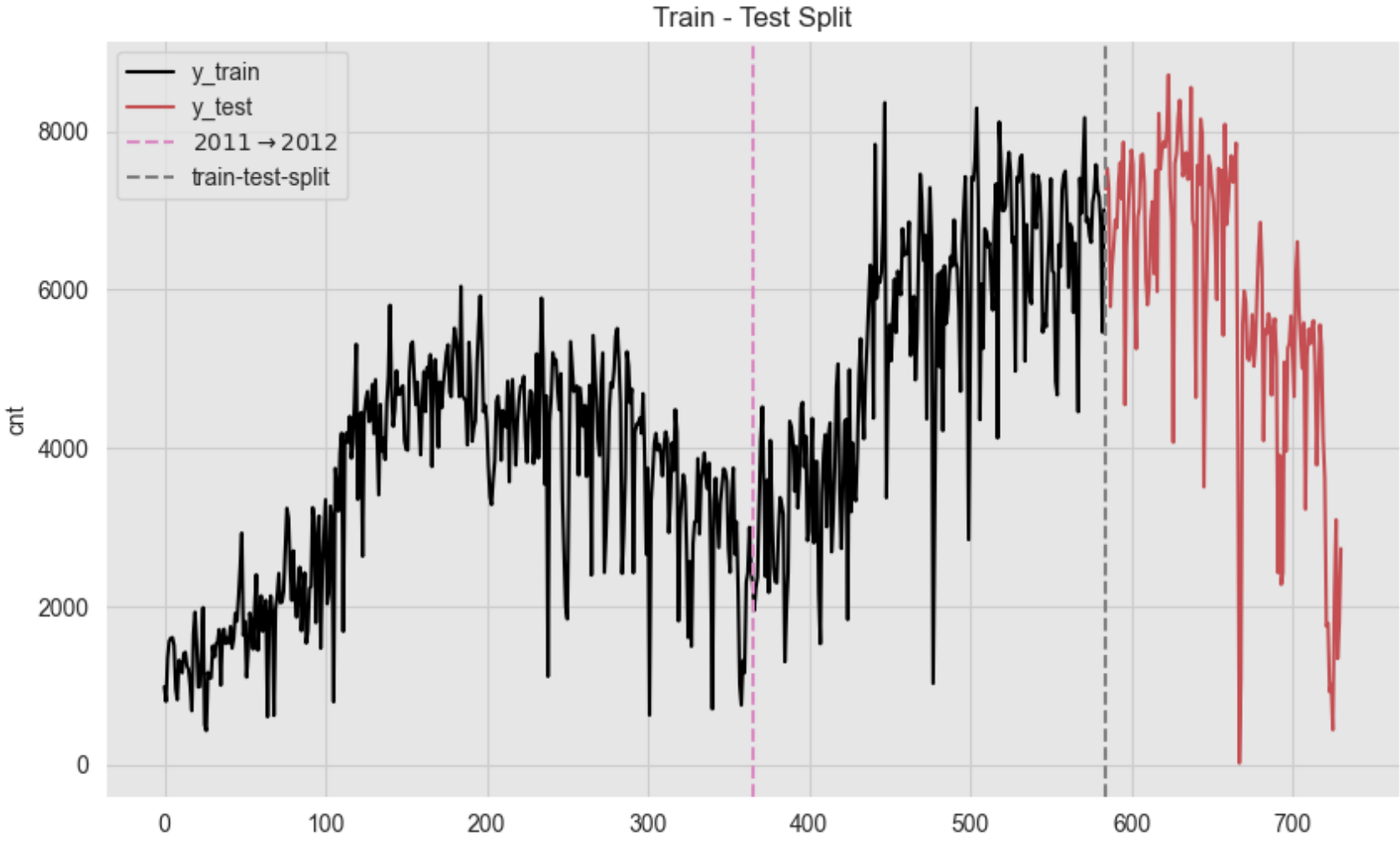

PyData Global

# Categorical Regressors




PyData Global

# Train-Test Split




PyData Global

# Models

## Two model flavours

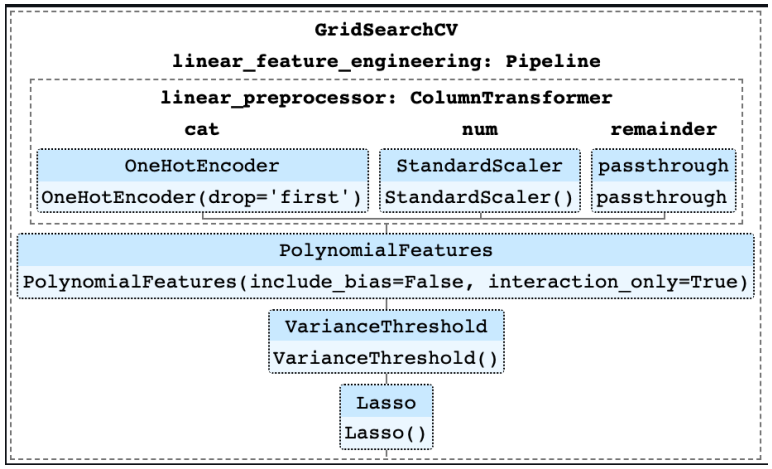



Figure: Lasso model with second order polynomial interactions.

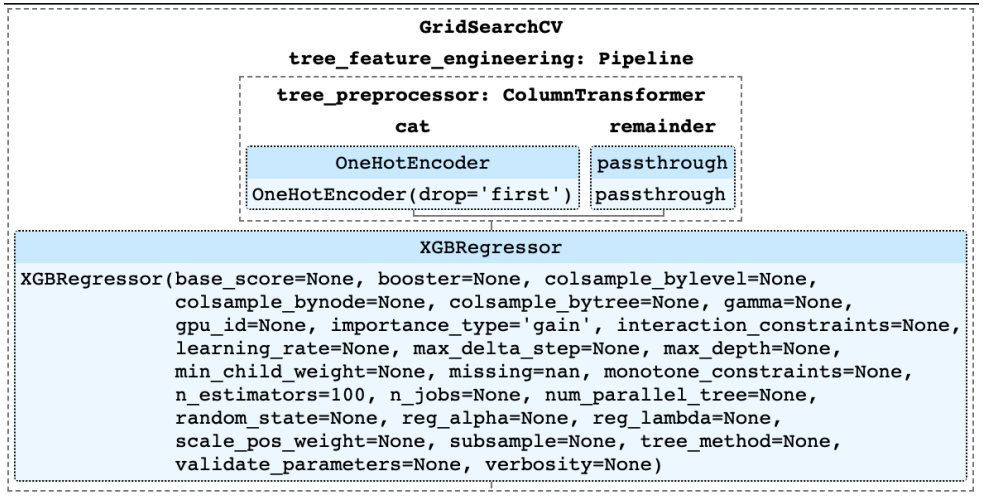
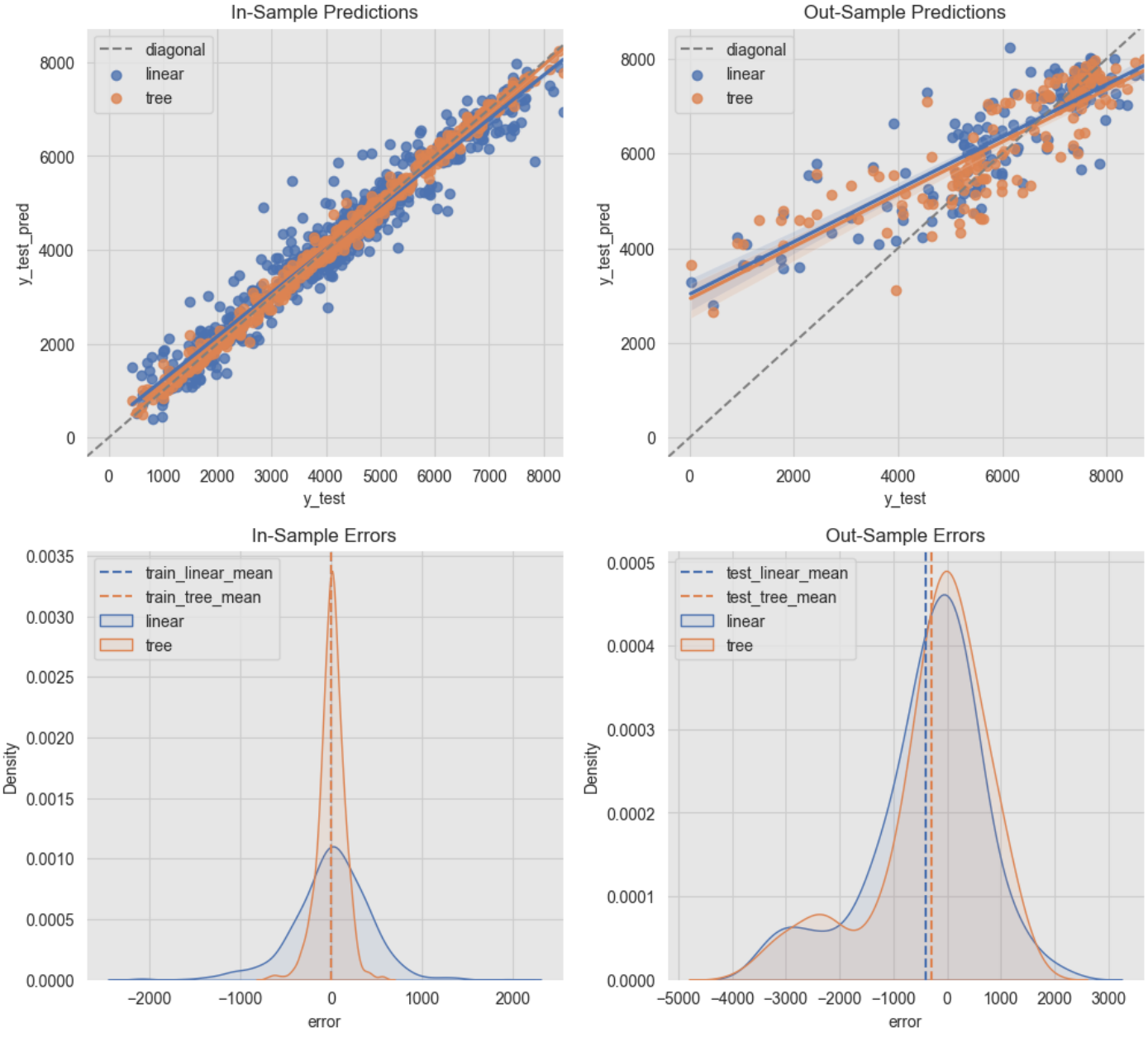



Figure: XGBoost regression model.

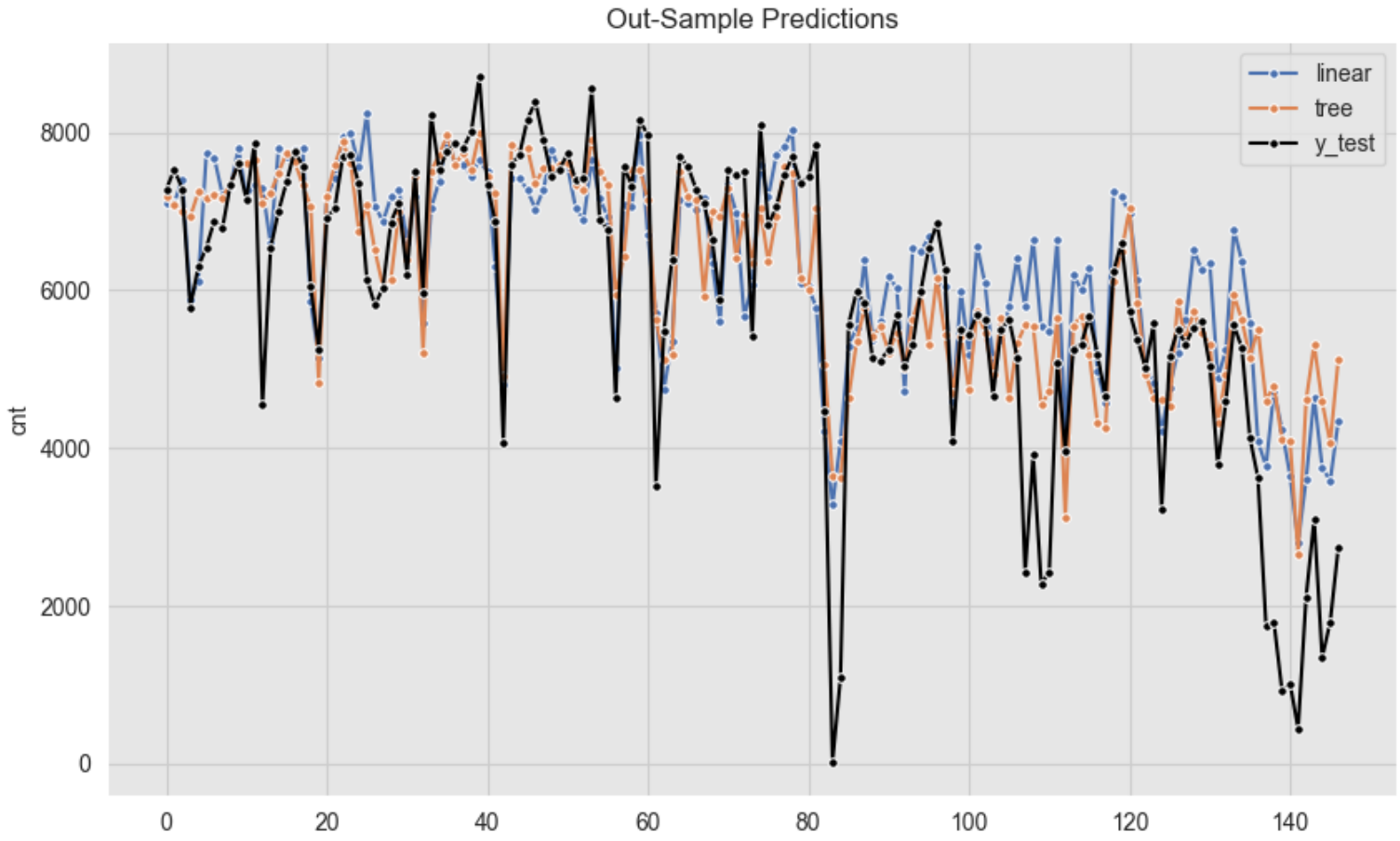
PyData Global

# Out of sample performance - Erros Distribution




PyData Global

# Out of sample performance - Predictions




PyData Global

# $\beta$ coeffiecients

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon, \quad \text{where} \quad \varepsilon \sim N(0, \sigma^2)$$

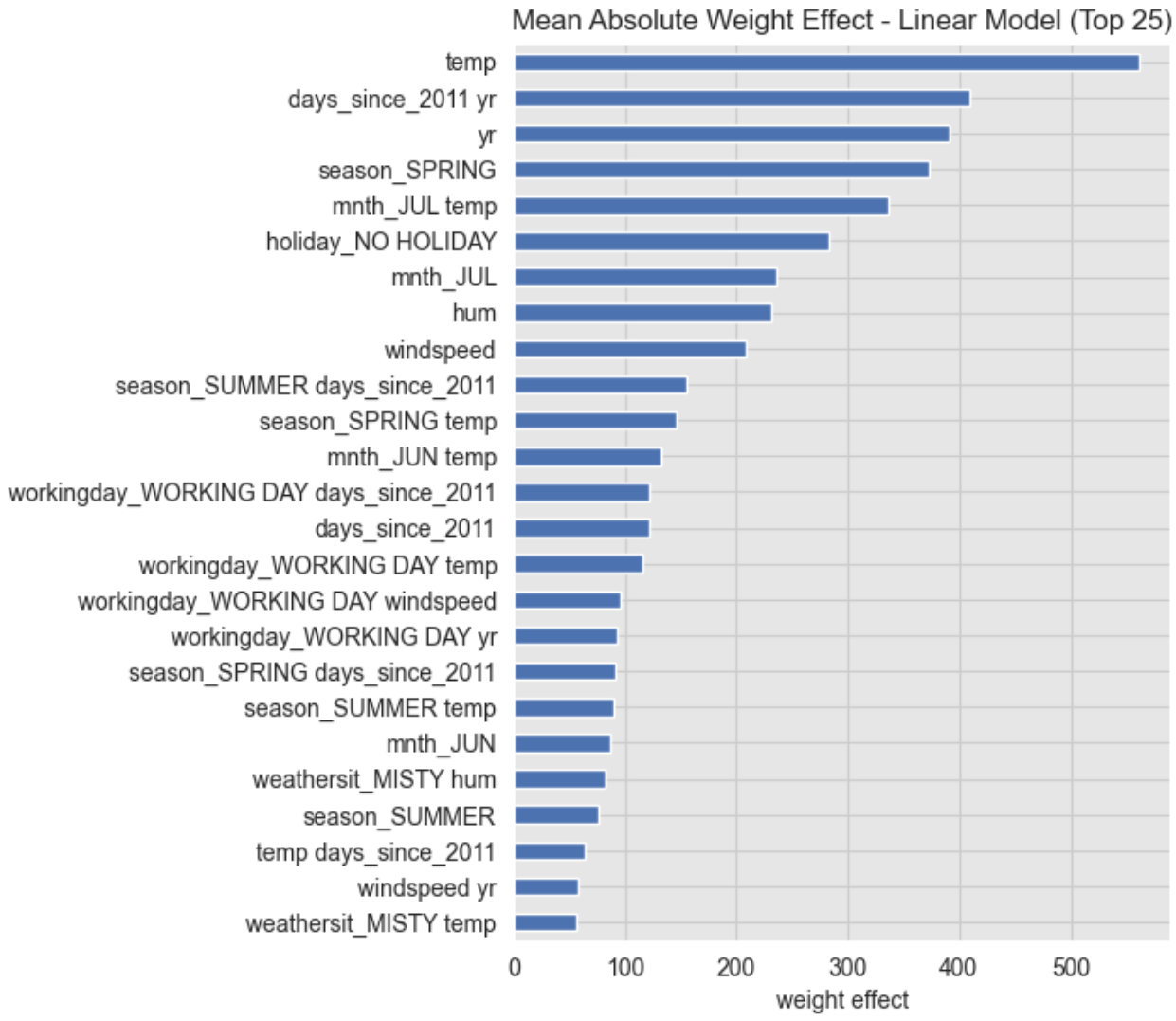
| | linear_features | coef_ | abs_coef_ |
|---|---|---|---|
| 0 | mnth_JUL temp | -2305.096894 | 2305.096894 |
| 1 | mnth_JUL | 2227.672335 | 2227.672335 |
| 2 | weathersit_RAIN/SNOW/STORM | -1710.469071 | 1710.469071 |
| 3 | mnth_JUN temp | -1299.644413 | 1299.644413 |
| 4 | season_SPRING | -1279.629779 | 1279.629779 |
| 5 | mnth_JUN | 845.229031 | 845.229031 |
| 6 | temp | 646.609622 | 646.609622 |
| 7 | mnth_AUG temp | -523.011653 | 523.011653 |
| 8 | season_SUMMER temp | 489.319256 | 489.319256 |
| 9 | weathersit_RAIN/SNOW/STORM temp | -482.660271 | 482.660271 |
| 10 | season_SPRING temp | 465.512410 | 465.512410 |
| 11 | days_since_2011 yr | 465.079169 | 465.079169 |
| 12 | weekday_SUN weathersit_RAIN/SNOW/STORM | -462.286059 | 462.286059 |
| 13 | season_SUMMER days_since_2011 | 454.137278 | 454.137278 |
| 14 | mnth_MAY temp | -445.268148 | 445.268148 |
| 15 | season_SUMMER weathersit_MISTY | -408.809531 | 408.809531 |
| 16 | mnth_MAY weathersit_MISTY | 404.790954 | 404.790954 |
| 17 | yr | 403.199142 | 403.199142 |
| 18 | season_SUMMER weathersit_RAIN/SNOW/STORM | -394.157306 | 394.157306 |
| 19 | mnth_DEZ temp | 363.222114 | 363.222114 |

PyData Global

# Weight Effects $\beta_i x_i$


PyData Global

# Weight Effects Importance $w_i = \frac{1}{n} \sum_{i=1}^{n} |\beta_i x_i|$




PyData Global

# Weight Effects: Temperature (z-transform)

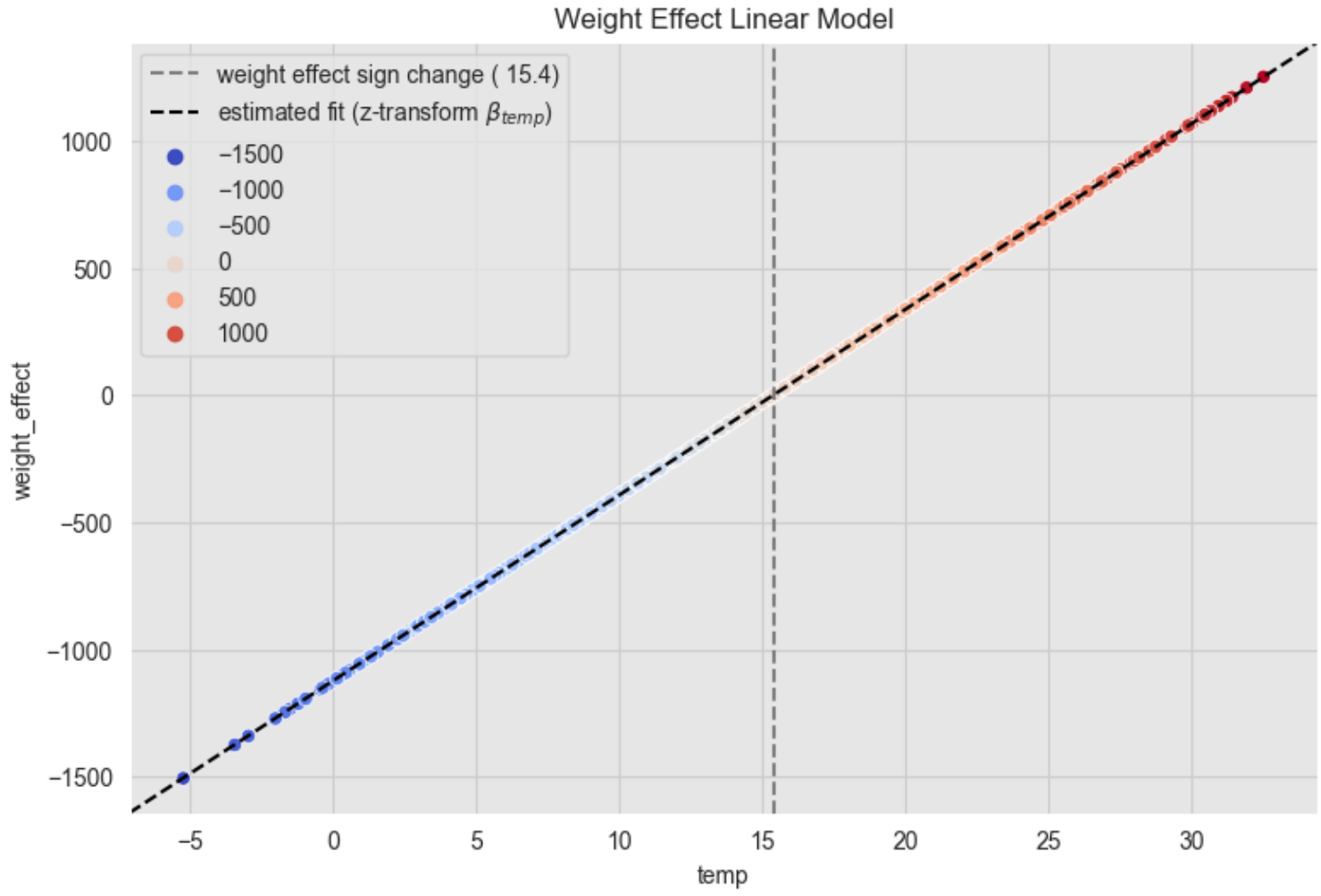



Figure: This plot just shows the effect of the linear term *temp* and not the interactions.

PyData Global

# Weight Effects: Interatctions

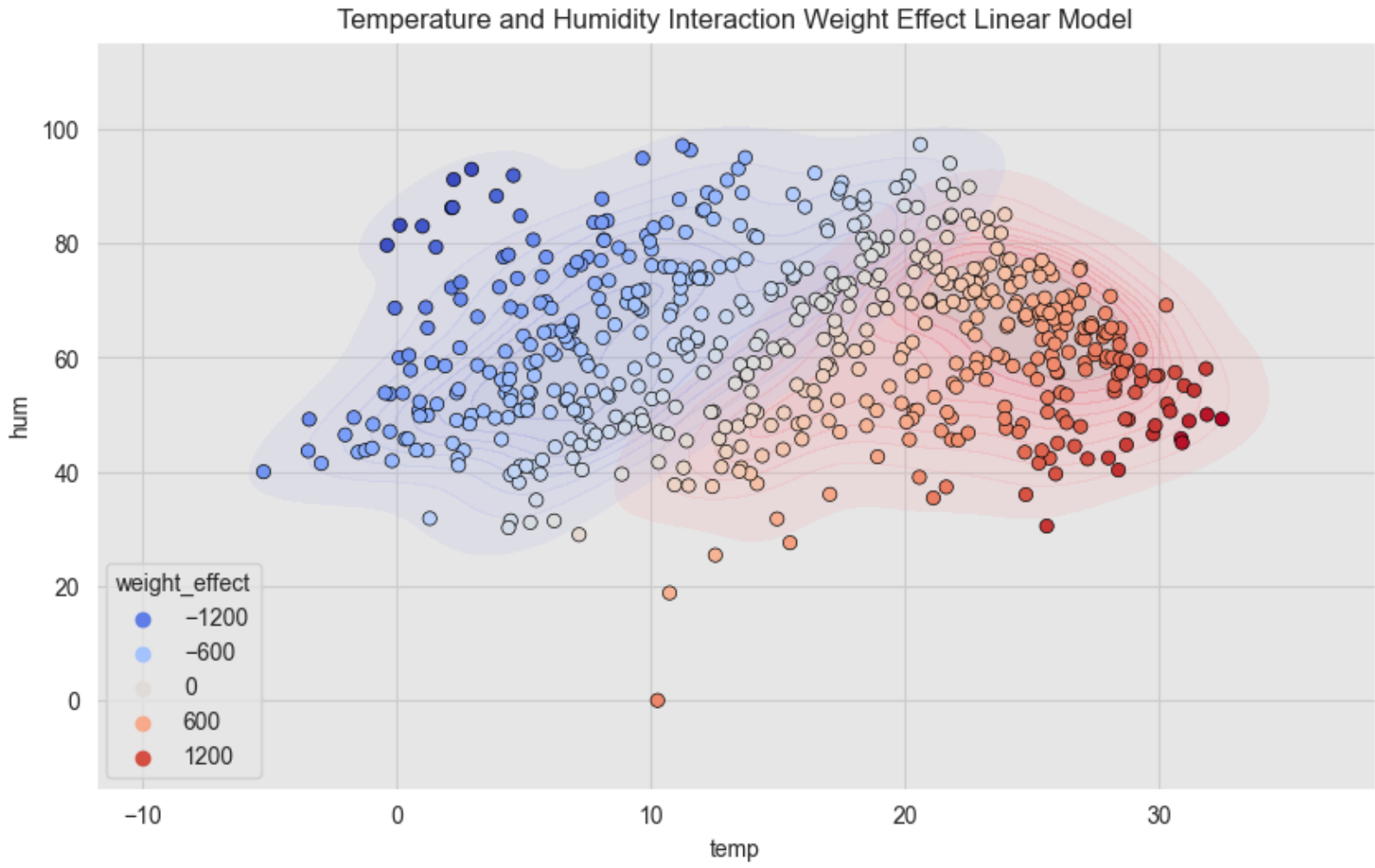
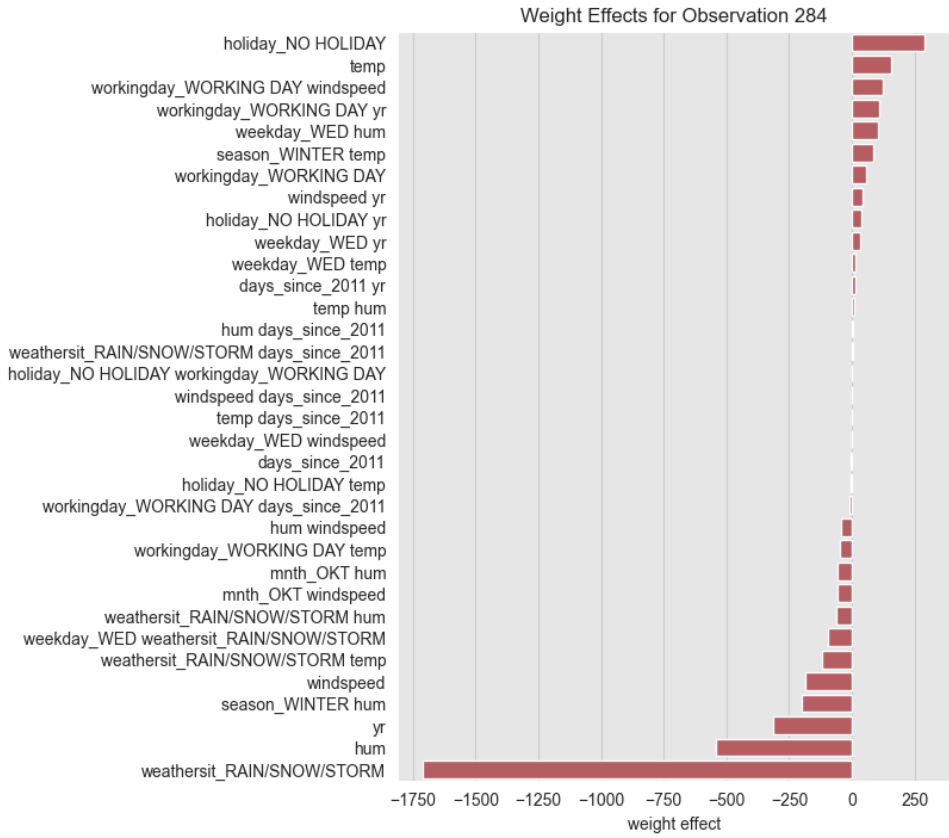
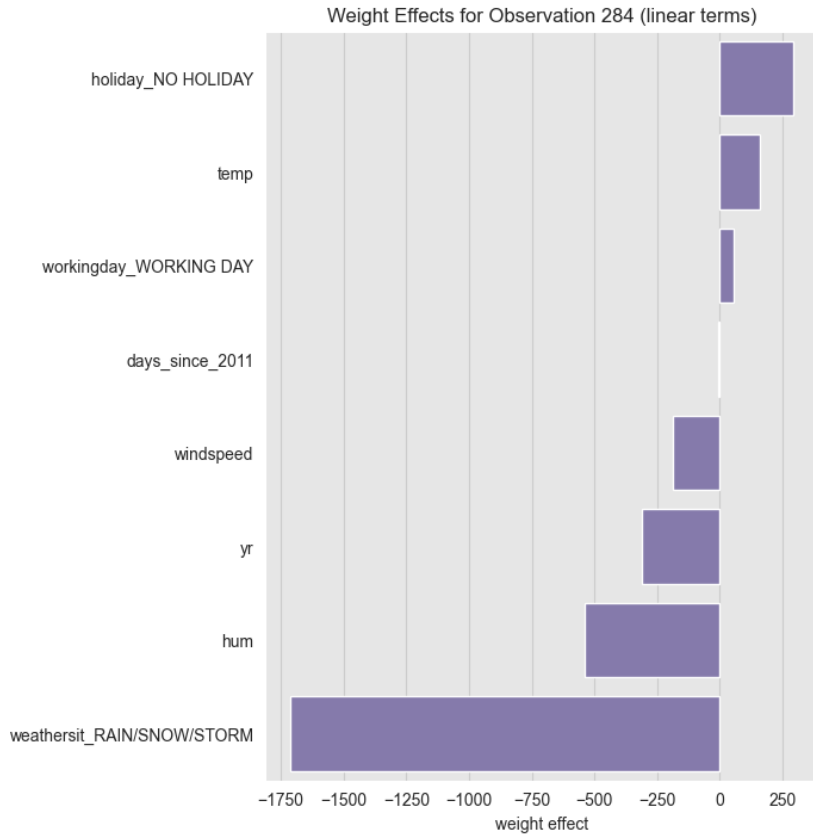



Figure: We can visualize the interaction between *temp* and *hum* by computing the total weight effect as $\beta_{temp}x_{temp} + \beta_{hum}x_{hum} + \beta_{temp \times hum}x_{temp}x_{hum}$.

PyData Global

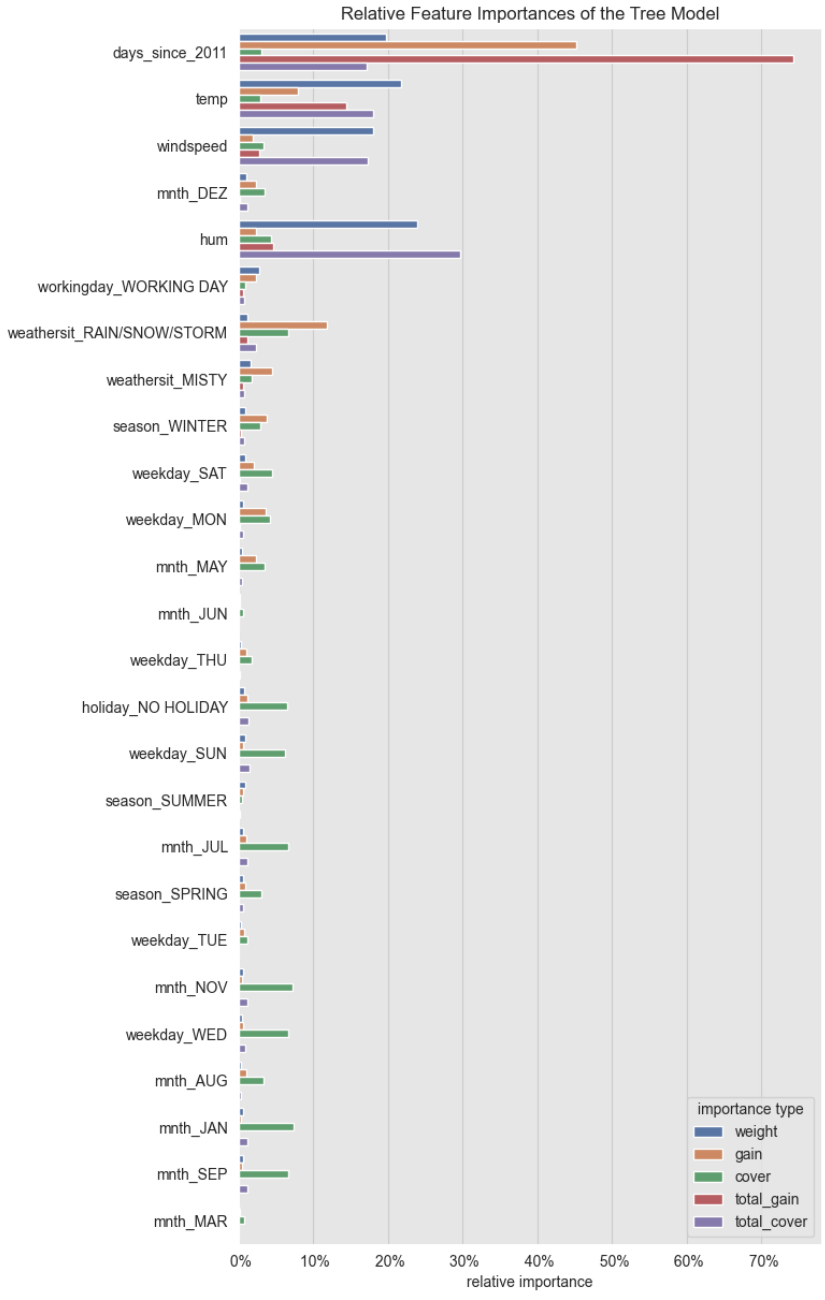
# Explaining Individual Predictions




PyData Global

# Feature Importance Metrics: XGBoost




PyData Global

# Partial Dependence Plot (PDP) & Individual Conditional Expectation (ICE) ([2, Section 5.1])

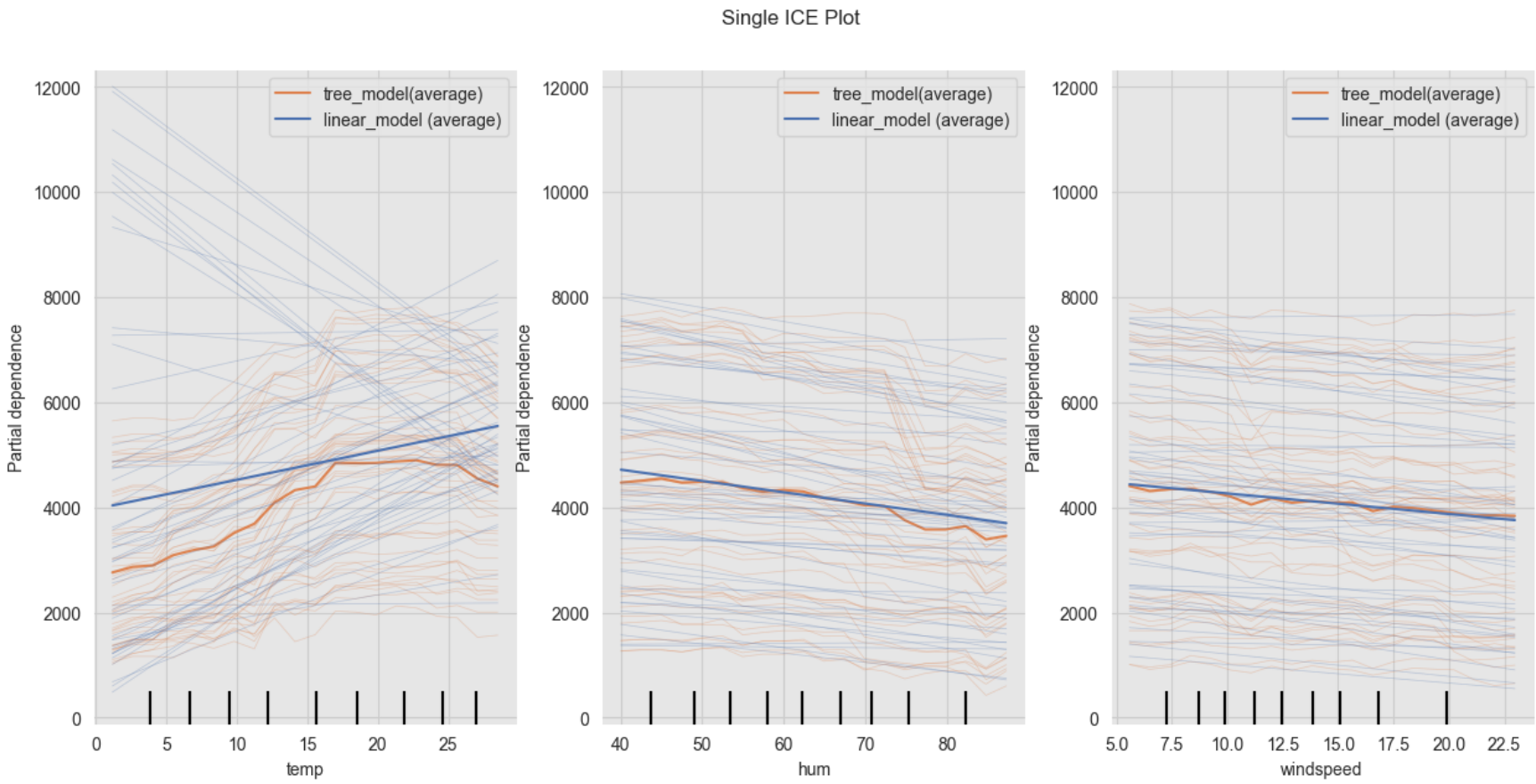
- Let $x_S$ be the features for which the partial dependence function should be plotted and $x_C$ be other features used in the machine learning model. One can estimate the *dependence function as*

$$\hat{f}_{x_S}(x_s) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(x_S, x_C^i)$$

  where $\hat{f}$ is the model prediction function, $x_C^i$ are actual feature values (not in $S$) and $n$ is the number points.
- Similar to a PDP, an individual conditional expectation (ICE) plot shows one line per instance. That is, for each instance in $\{(x_S^i, x_C^i)\}_{i=1}^n$, we plot $\hat{f}_S$ as a function of $x_S^i$ while leaving $x_C^i$ fixed.

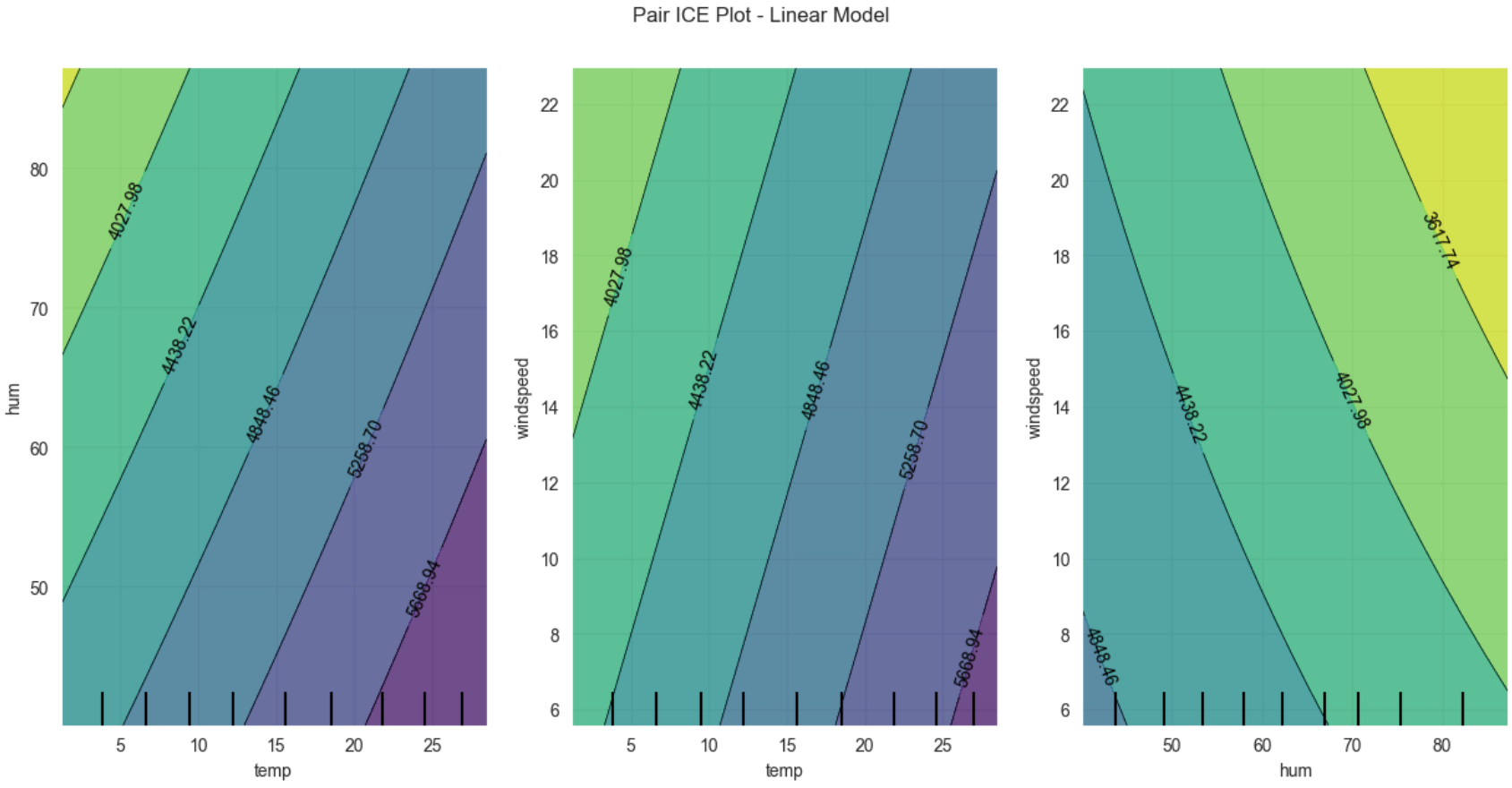
PyData Global

# PDP & ICE Examples (1D)




PyData Global

# PDP & ICE Examples (2D)




| | linear_features | coef_ | abs_coef_ |
|---|---|---|---|
| 6 | temp | 646.609622 | 646.609622 |
| 29 | hum | -281.701209 | 281.701209 |
| 32 | windspeed | -264.190697 | 264.190697 |
| 114 | hum windspeed | -31.993914 | 31.993914 |
| 130 | temp hum | 15.127133 | 15.127133 |
| 155 | temp windspeed | -0.000000 | 0.000000 |

PyData Global

# Permutation Importance

Measures the increase in the prediction error of the model after we permuted the feature's values, which breaks the relationship between the feature and the true outcome ([2, Section 5.6]).

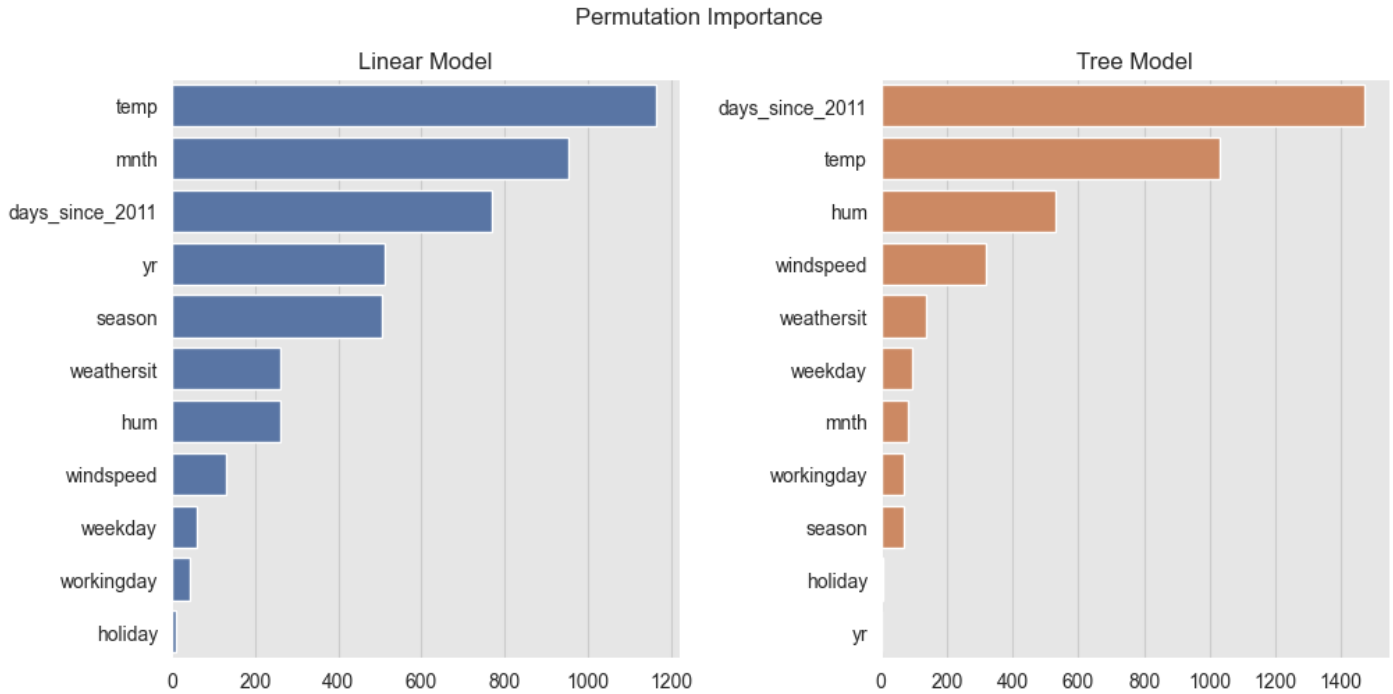
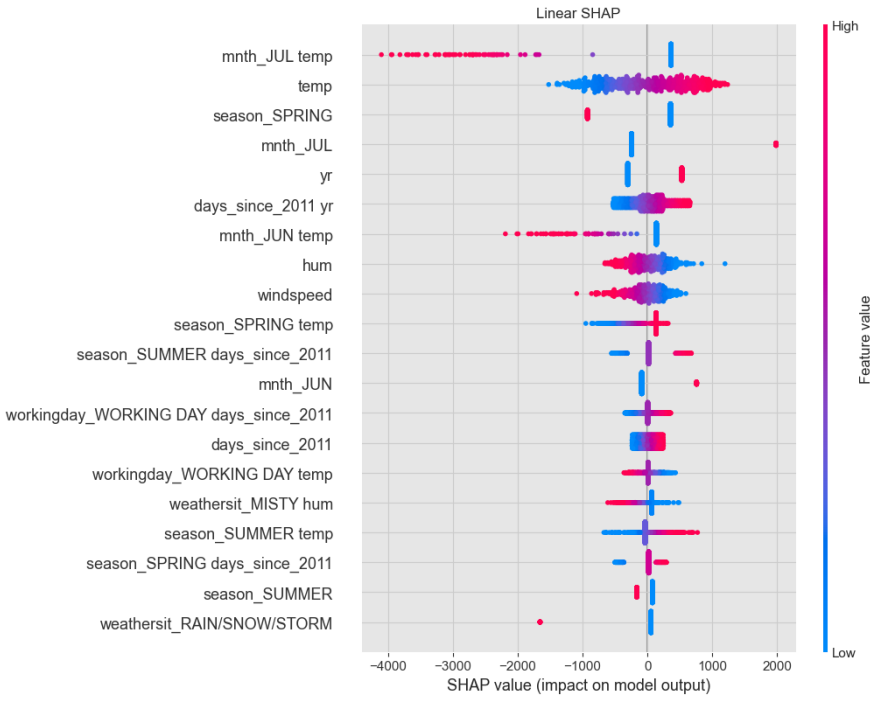
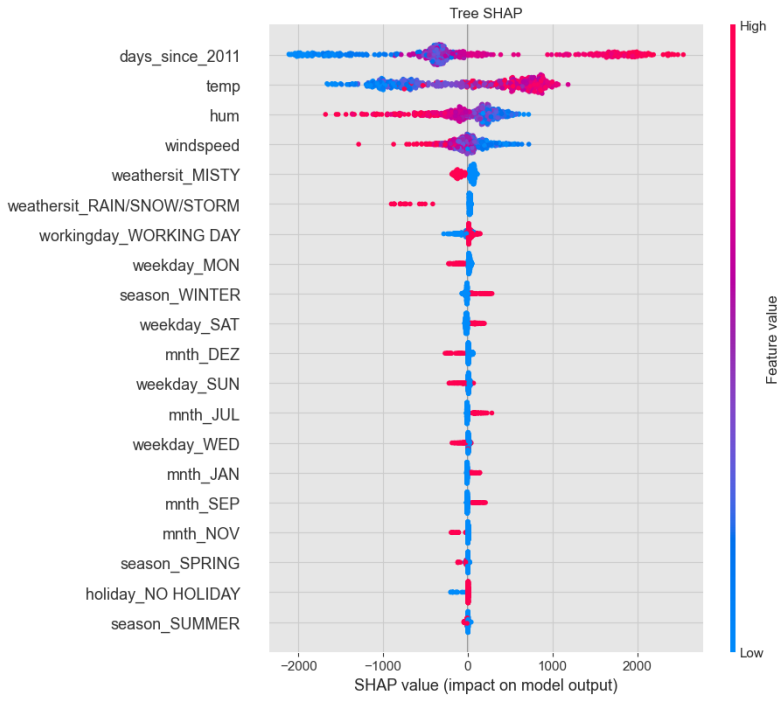



Figure: The permutation importance for these two models have *days_since_2011* and *temp* on their top 3 ranking, which partially explain the trend and seasonality components respectively (see [2, Figure 5.32]).

PyData Global

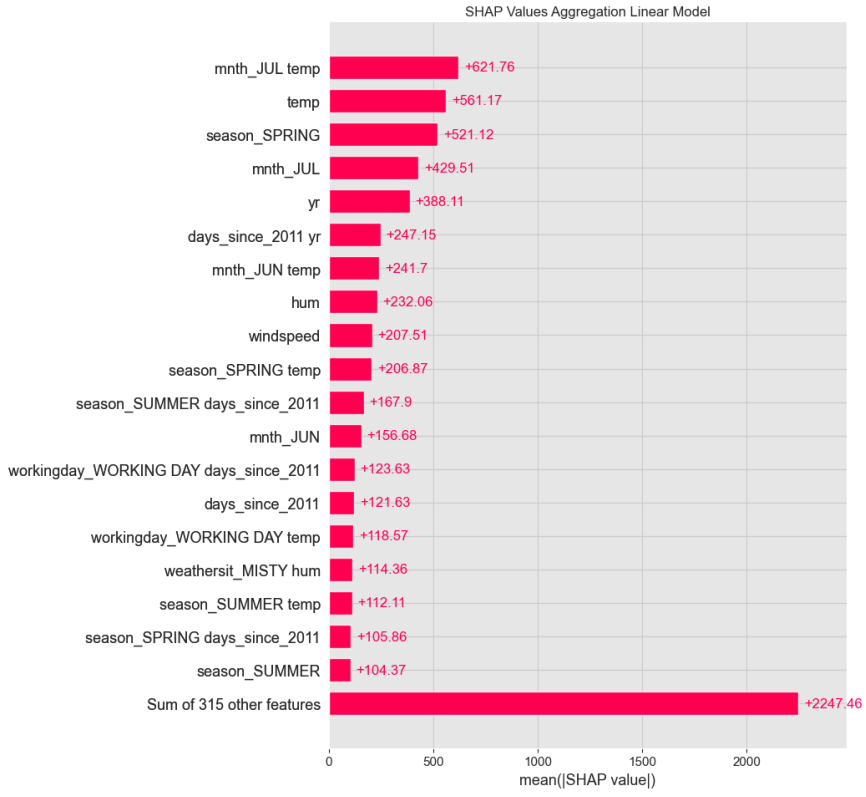
# SHAP Values

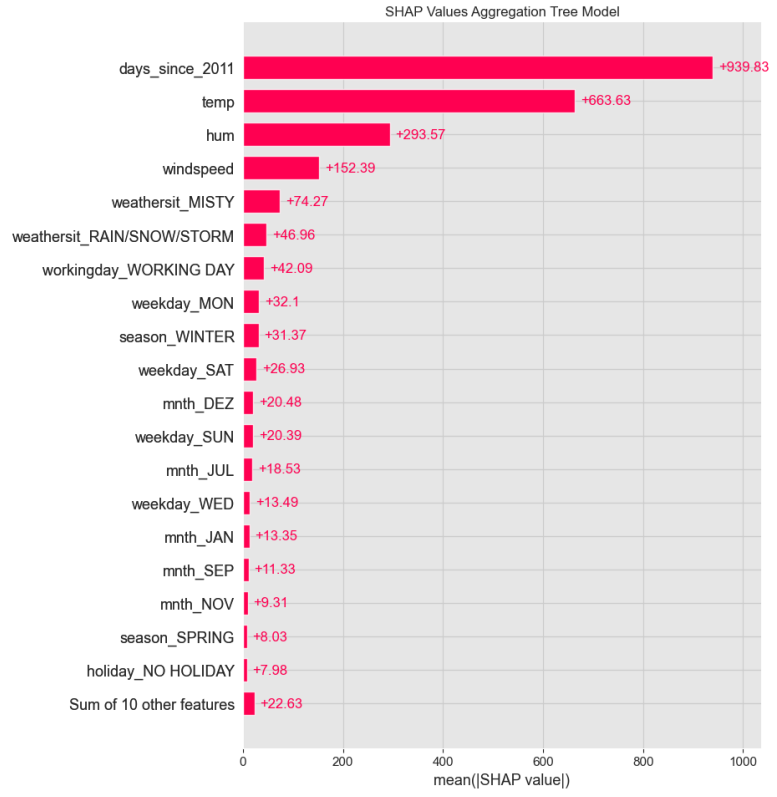



PyData Global

# Mean Abs SHAP Values




PyData Global

# SHAP Values: Temperature

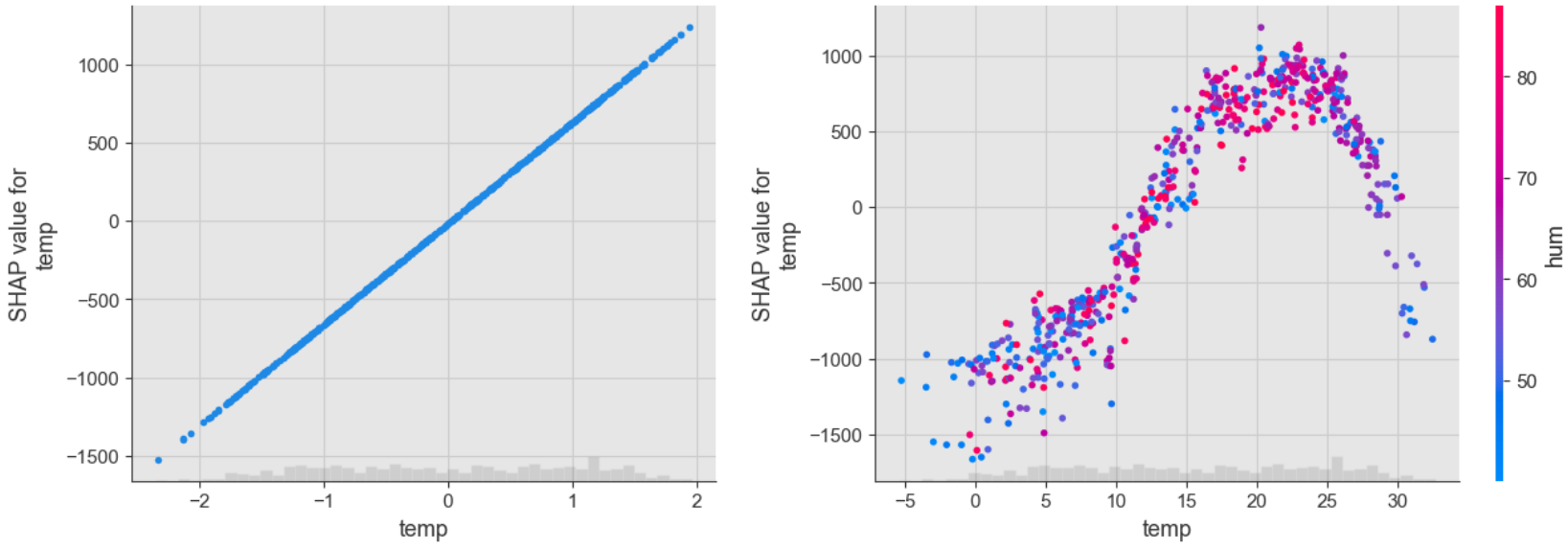
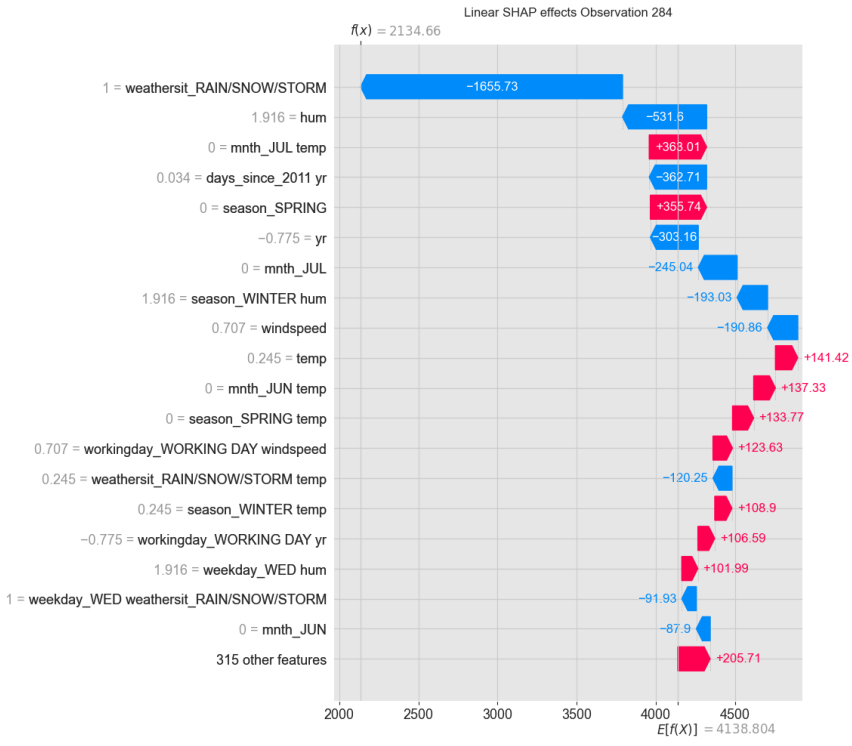



Figure: ...

PyData Global

# SHAP Values: Observation 284



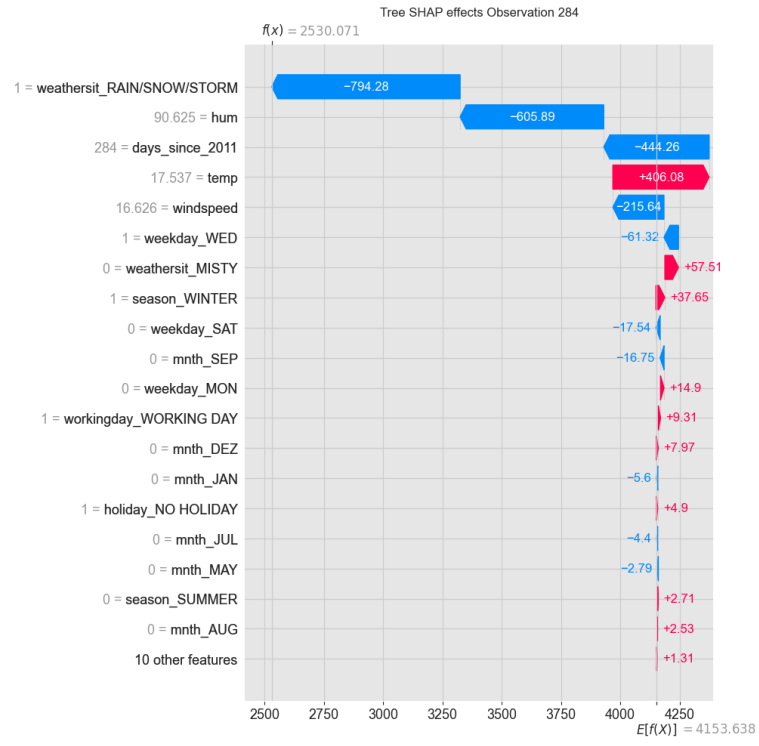




Figure: ...

PyData Global

# References I


[1] Serg Masís.

*Interpretable Machine Learning with Python*.

Packt Publishing, 2021.

https://github.com/PacktPublishing/Interpretable-Machine-Learning-with-Python.

[2] Christoph Molnar.

*Interpretable Machine Learning*.

2019.

https://christophm.github.io/interpretable-ml-book/.

[3] Juan Orduz.

Exploring tools for interpretable machine learning.

https://juanitorduz.github.io/interpretable_ml/, Jul 2021.

# Thank You!

## Contact

- https://juanitorduz.github.io
- github.com/juanitorduz
- juanitorduz
- juanitorduz@gmail.com

PyData Global