# Driverless AI Experiment: wm-default-AllCovariate

Generated by: yuki.shimada@h2o.ai

Generated on: Wed Feb 26 04:20:48 2025

[Driverless AI Experiment: wm-default-AllCovariate 1](#_Toc55907577)

[Experiment Overview 1](#_Toc55907578)

[Data Overview 1](#_Toc55907579)

[Methodology 1](#_Toc55907580)

[Data Sampling 1](#_Toc55907581)

[Validation Strategy 2](#_Toc55907582)

[Model Tuning 2](#_Toc55907583)

[Feature Evolution 2](#_Toc55907584)

[Feature Transformations 2](#_Toc55907585)

[Final Model 2](#_Toc55907586)

[Alternative Models 2](#_Toc55907587)

[Deployment 2](#_Toc55907588)

[Partial Dependence Plots 2](#_Toc55907589)

[Shapley Values 2](#_Toc55907590)

[Appendix 3](#_Toc55907593)

## Experiment Overview

Driverless AI built 1 XGBoostGBMModel to predict Weekly\_Sales given 10 original features from the input dataset walmart\_ts\_6\_fcst\_grp\_train.csv. This regression experiment completed in 19 minutes and 18 seconds (0:19:18), using 2 of the 10 original features, and 38 of the 38 engineered features.

### Performance

|  |  |
| --- | --- |
| **Dataset** | **RMSE** |
| Internal Validation | 1747.488 |
| Test Data | 1706.288 |

### Driverless AI Settings

|  |  |  |  |
| --- | --- | --- | --- |
| **Dial Settings** | **Description** | **Setting Value** | **Range of Possible Values** |
| Accuracy | Controls accuracy needs of the model | 7 | 1-10 |
| Time | Controls duration of the experiment | 5 | 1-10 |
| Interpretability | Controls complexity of the model | 7 | 1-10 |

### System Specifications

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Address** | **System** | **System Memory** | **CPUs** | **GPUs** |
| http://127.0.0.1:12345 | Linux | 248 GB | 32 | 0 |

### Versions

|  |  |
| --- | --- |
|  |  |
| Driverless AI version | 1.10.7.2 |
| h2o4gpu version | 0.4.2 |
| h2o\_mli version | 1.10.29 |
| mojo2\_runtime version | 2.8.5 |
| daimojo version | 2.8.5 |
| daimojo version\_ppc64le | 2.8.1 |
| procsy version | 0.9.0 |
| pydatatable version | 1.1.0a2217 |
| vis\_data\_server version | 2.1.4 |

## Data Overview

This section provides information on the datasets used for the experiment.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **data** | **file path** | **file size** | **number of rows** | **number of columns** |
| training | /dai-data/yuki.shimada@h2o.ai/b0f4f98c-f3f3-11ef-a69c-4e030d02e593/walmart\_ts\_6\_fcst\_grp\_train.csv.1740541344.4240518.bin | 52.0 KiB | 702 | 11 |
| validation | Not provided | None | None | None |
| testing | /dai-data/yuki.shimada@h2o.ai/b0bcca8a-f3f3-11ef-a69c-4e030d02e593/walmart\_ts\_6\_fcst\_grp\_test.csv.1740541344.0607128.bin | 3.8 KiB | 36 | 11 |

### Training Data

The training data consists of only numeric columns.

The summary of the columns is shown below:

#### Numeric Columns

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **name** | **logical\_type** | **storage\_type** | **min** | **mean** | **max** | **std** | **unique** | **freq of mode** |
| Store | numeric, categorical, catlabel, ohe\_categorical, raw | int | 1.000 | 12.667 | 31.000 | 10.085 | 6 | 117 |
| Dept | numeric, categorical, catlabel, ohe\_categorical, raw | int | 4.000 | 9.833 | 13.000 | 3.438 | 3 | 351 |
| Date | catlabel, date, raw, time\_column | time [%Y/%m/%d] | 2010/02/05 | nan | 2012/04/27 | nan | 117 | 6 |
| Weekly\_Sales | N/A | real | 20,190.540 | 40,117.380 | 72,179.920 | 9,670.895 | 702 | 1 |
| MarkDown1 | numeric, categorical, catlabel | real | -1.000 | 1,978.312 | 60,394.730 | 6,000.891 | 151 | 552 |
| MarkDown2 | numeric, categorical, catlabel | real | -265.760 | 1,386.535 | 89,121.940 | 6,444.609 | 144 | 558 |
| MarkDown3 | numeric, categorical, catlabel | real | -1.000 | 701.249 | 109,030.750 | 6,962.930 | 148 | 555 |
| MarkDown4 | numeric, categorical, catlabel | real | -1.000 | 951.741 | 57,817.560 | 4,587.991 | 151 | 552 |
| MarkDown5 | numeric, categorical, catlabel | real | -1.000 | 1,408.077 | 31,844.200 | 3,647.650 | 151 | 552 |
| sample\_weight | N/A | int | 1.000 | 1.308 | 5.000 | 1.067 | 2 | 54 |

#### Boolean Columns

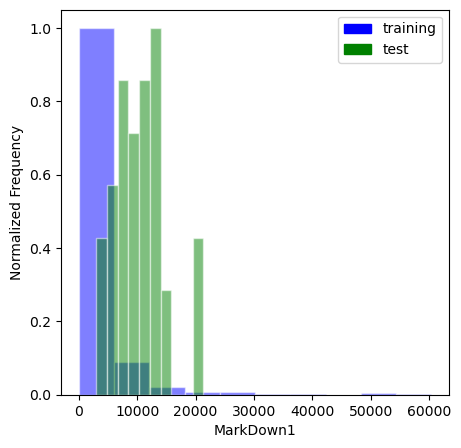
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **name** | **logical\_type** | **storage\_type** | **min** | **mean** | **max** | **std** | **freq of max value** |
| IsHoliday | numeric | bool | False | 0.0769 | True | 0.2667 | 54 |

#### Shifts Detected

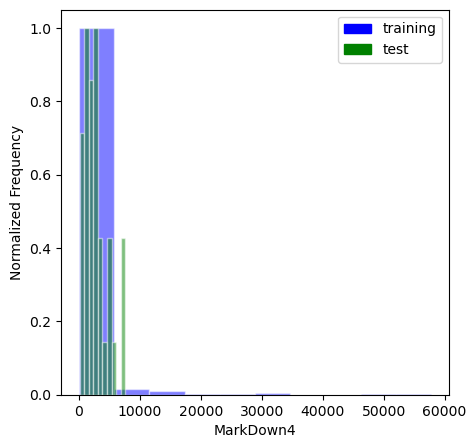
Driverless AI can perform shift detection between the training, validation, and testing datasets. It does this by training a binomial model to predict which dataset a record belongs to. For example, it may find that it is able to separate the training and testing data with an AUC of 0.8 using only the column: C1 as the predictor. This indicates that there is some sort of drift in the distribution of C1 between the training and testing data.

For this experiment, Driverless AI checked the train and test data for any shift in distribution and found the following significant differences:

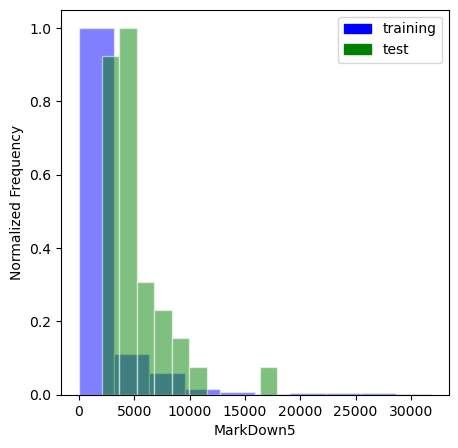
* Significant difference detected between training and test data distribution for feature <<<MarkDown1>>> (AUC: 0.956).



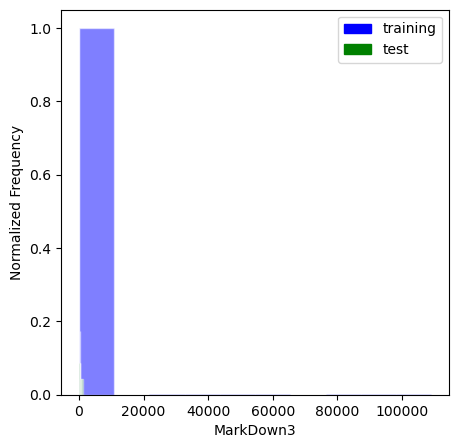
* Significant difference detected between training and test data distribution for feature <<<MarkDown4>>> (AUC: 0.928).



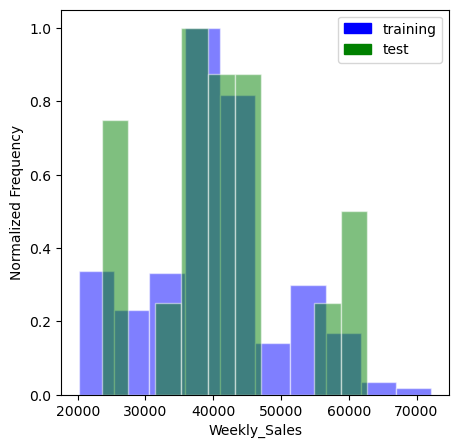
* Significant difference detected between training and test data distribution for feature <<<MarkDown5>>> (AUC: 0.915).



* Significant difference detected between training and test data distribution for feature <<<MarkDown3>>> (AUC: 0.886).



* Significant difference detected between training and test data distribution for feature <<<Weekly\_Sales>>> (AUC: 0.554).



## Methodology

This section describes the experiment methodology.

### Assumptions and Limitations

Driverless AI trains all models based on the training data provided (in this case: *walmart\_ts\_6\_fcst\_grp\_train.csv*). It is the assumption of Driverless AI that this dataset is representative of the data that will be seen when scoring.

Driverless AI may perform shift detection between the train and test data. If a shift in distribution is detected, this may indicate that the data that will be used for scoring may have distributions not represented in the training data.

For this experiment, Driverless AI performed shift detection and found significant differences described below:

|  |  |  |
| --- | --- | --- |
| **shift\_col** | **shift\_data\_first** | **shift\_data\_second** |
| MarkDown1 | training | test |
| MarkDown4 | training | test |
| MarkDown5 | training | test |
| MarkDown3 | training | test |
| Weekly\_Sales | training | test |

### Experiment Pipeline

For this experiment, Driverless AI performed the following steps to find the optimal final model:



The steps in this pipeline are described in more detail below:

* **Ingest Data**

detected column types

* **Feature Preprocessing**

turned raw features into numerical values

* **Model and Feature Tuning**

This stage combines random hyperparameter tuning with feature selection and generation. Features in each iteration are updated using variable importance from the previous iteration as a probabilistic prior to decide what new features to create. The best performing model and features are then passed to the feature evolution stage.

found the optimal parameters for lightgbm, constant and xgboost models by training models with different parameters

the best parameters are those that generate the least **RMSE** on the internal validation data

81 models trained and scored to evaluate features and model parameters

* **Feature Evolution**

This stage uses a genetic algorithm to find the best set of model parameters and feature transformations to be used in the final model.

found the best representation of the data for the final model training by creating and evaluating **38** features over **63** iterations

trained and scored 648 models to further evaluate engineered features

* **Final Model**

created the best model from the feature engineering iterations

no stacked ensemble is done because a time column was provided

* **Create Scoring Pipeline**

created and exported the MOJO and Python scoring pipeline

MOJO Scoring Pipeline: yuki.shimada@h2o.ai/h2oai\_experiment\_cf326492-f3f4-11ef-a69c-4e030d02e593/mojo\_pipeline/mojo.zip

Python Scoring Pipeline: yuki.shimada@h2o.ai/h2oai\_experiment\_cf326492-f3f4-11ef-a69c-4e030d02e593/scoring\_pipeline/scorer.zip

**Models for Optimization**

Driverless AI trained models throughout the experiment to determine the best parameters, model dataset, and optimal final model. The stages are described below:

|  |  |  |
| --- | --- | --- |
| **Driverless AI Stage** | **Timing (seconds)** | **Number of Models** |
| Data Preparation | 5.68 | 0 |
| Model and Feature Tuning | 177.56 | 81 |
| Feature Evolution | 865.35 | 648 |
| Final Pipeline Training | 44.56 | 1 |

### Experiment Settings

Below are the settings selected for the experiment by yuki.shimada@h2o.ai. The Defined Parameters represent the high-level parameters.

**Defined Parameters**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| is\_classification | False |
| enable\_gpus | False |
| seed | False |
| accuracy | 7 |
| time | 5 |
| interpretability | 7 |
| num\_prediction\_periods | 6 |
| num\_gap\_periods | 0 |
| is\_timeseries | True |
| is\_image | False |

These Accuracy, Time, and Interpretability settings map to the following internal configuration of the Driverless AI experiment:

|  |  |
| --- | --- |
| **Internal Parameter** | **Value** |
| data filtered | False |
| tune target transform | True |
| number of feature engineering iterations | 50 |
| number of models trained per iteration | 8 |
| early stopping rounds | 10 |
| monotonicity constraint | True |
| number of model tuning model combinations | 12 |
| number of base learners in ensemble | 0 |
| time column | Date |
| time group columns | ['Date', 'Dept', 'Store'] |
| time period | week |
| number of prediction periods | 6 |
| number of gap periods | 0 |

#### Details

* **data filtered**: Driverless AI may filter the training data depending on the number of rows and the Accuracy setting.

for this experiment, the training data was not filtered.

* **tune target transform**: whether Driverless AI evaluated the model performance if the target was transformed.

eg: the model performance may be better by predicting the log of the target column instead of the raw target column

* **number of feature engineering iterations**: the number of iterations performed of feature engineering.
* **number of models evaluated per iteration**: for each feature engineering iteration, Driverless AI trains multiple models, if applicable. Each model is trained with a different set of predictors or features. The goal of this step is to determine which types of features lead to the least RMSE .
* **early stopping rounds**: if Driverless AI does not see any improvement after 10 iterations of feature engineering, the feature engineering step is automatically stopped.
* **monotonicity constraint**: if enabled, the models will only have monotone relationships between the predictors and target variable. This is off by default given no target variable involved.
* **number of model tuning combinations**: the number of model tuning combinations evaluated to determine the optimal model settings for the constant, xgboost and lightgbm models.
* **number of base learners in ensemble**: the number of base models used to create the final ensemble.
* **time column**: the column that provides the time column. If a time column is provided, feature engineering and model validation will respect the causality of time. If the time column is turned off, no time order is used for modeling and data may be shuffled randomly (any potential temporal causality will be ignored). This is off by default for anomaly detection.
* **time group columns**: the columns that make up the time series groups.
* **time period**: the periodicity found in the dataset.
* **number of prediction periods**: the number of periods you want to predict in advance.
* **number of gap periods**: the gap between the data available and the forecast period desired.

## Data Sampling

In Driverless AI, data sampling is a pre-processing step that is done before model training begins; it is not related to sampling done during model training. Driverless AI does not perform data sampling unless the dataset is big or highly imbalanced. Whether a dataset is considered big depends on the experiment’s accuracy setting and the statistical\_threshold\_data\_size\_large config.toml parameter.

Driverless AI did not perform any down sampling of the data.

## Validation Strategy

Driverless AI automatically split the data into training and validation data, ordering the data by Date. The model was tuned to predict 6 weeks with no gap between training and forecasting.

### Time Based Validation Splits for Model Tuning and Feature Evolution

Blue color represents training data split while gray color represents validation data split. Following data splits were used:

**evolution\_fold\_1\_train**

* starts 2010-02-05 00:00:00
* ends 2012-02-03 00:00:00

**evolution\_fold\_1\_valid**

* starts 2012-02-10 00:00:00
* ends 2012-04-27 00:00:00

**evolution\_fold\_2\_train**

* starts 2010-02-05 00:00:00
* ends 2011-11-11 00:00:00

**evolution\_fold\_2\_valid**

* starts 2011-11-18 00:00:00
* ends 2012-02-03 00:00:00

**evolution\_fold\_3\_train**

* starts 2010-02-05 00:00:00
* ends 2011-08-19 00:00:00

**evolution\_fold\_3\_valid**

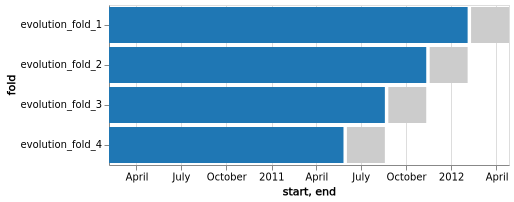
* starts 2011-08-26 00:00:00
* ends 2011-11-11 00:00:00

**evolution\_fold\_4\_train**

* starts 2010-02-05 00:00:00
* ends 2011-05-27 00:00:00

**evolution\_fold\_4\_valid**

* starts 2011-06-03 00:00:00
* ends 2011-08-19 00:00:00



### Time Based Back Testing of Final Pipeline on Training Data

Blue color represents training data split while gray color represents validation data split. Following data splits were used:

**final\_pipeline\_backtest\_fold\_1\_train**

* starts 2010-02-05 00:00:00
* ends 2012-02-03 00:00:00

**final\_pipeline\_backtest\_fold\_1\_test**

* starts 2012-02-10 00:00:00
* ends 2012-04-27 00:00:00

**final\_pipeline\_backtest\_fold\_2\_train**

* starts 2010-02-05 00:00:00
* ends 2011-11-11 00:00:00

**final\_pipeline\_backtest\_fold\_2\_test**

* starts 2011-11-18 00:00:00
* ends 2012-02-03 00:00:00

**final\_pipeline\_backtest\_fold\_3\_train**

* starts 2010-02-05 00:00:00
* ends 2011-08-19 00:00:00

**final\_pipeline\_backtest\_fold\_3\_test**

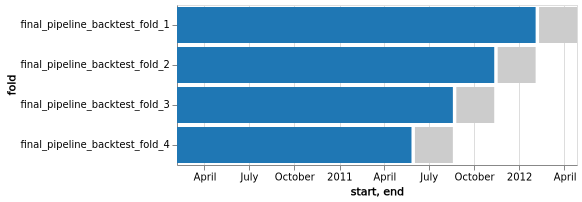
* starts 2011-08-26 00:00:00
* ends 2011-11-11 00:00:00

**final\_pipeline\_backtest\_fold\_4\_train**

* starts 2010-02-05 00:00:00
* ends 2011-05-27 00:00:00

**final\_pipeline\_backtest\_fold\_4\_test**

* starts 2011-06-03 00:00:00
* ends 2011-08-19 00:00:00



### Training Data for Final Pipeline

Blue color represents training data split while gray color represents validation data split. Following data splits were used:

**final\_pipeline\_train**

* starts 2010-02-05 00:00:00
* ends 2012-04-27 00:00:00



## Model Tuning

The table below shows the score and training time of the lightgbm, constant and xgboost models evaluated by Driverless AI. The table shows the top 10 parameter tuning models evaluated, ordered based on a combination of least score and lowest training time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **job order** | **booster** | **nfeatures** | **training times** | **scores (RMSE)** |
| 5 | gbtree | 44 | 6.7641 | 1723.1128 |
| 9 | gbtree | 82 | 6.3002 | 1989.4866 |
| 7 | gbtree | 82 | 6.3359 | 1989.4866 |
| 1 | gbtree | 37 | 4.4647 | 2089.4375 |
| 13 | lightgbm | 82 | 2.6204 | 2168.9126 |
| 14 | lightgbm | 82 | 2.7091 | 2168.9126 |
| 0 | lightgbm | 44 | 2.0939 | 2178.3149 |
| 11 | lightgbm | 44 | 2.3647 | 2179.605 |
| 10 | lightgbm | 44 | 2.8194 | 2225.6084 |
| 2 | lightgbm | 44 | 2.7761 | 2244.6536 |

More detailed information on the parameters evaluated for each algorithm is shown below.

### lightgbm tuning

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **tree method** | **grow policy** | **max depth** | **max leaves** | **colsample bytree** | **subsample** | **nfeatures** | **training times** | **scores (RMSE)** |
|  | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 82 | 2.6204 | 2168.9126 |
|  | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 82 | 2.7091 | 2168.9126 |
|  | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 44 | 2.0939 | 2178.3149 |
|  | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 44 | 2.3647 | 2179.605 |
|  | depthwise | 8.0 | 256.0 | 0.8 | 0.7 | 44 | 2.8194 | 2225.6084 |
|  | depthwise | 8.0 | 256.0 | 0.8 | 0.7 | 44 | 2.7761 | 2244.6536 |
|  | lossguide | 8.0 | 2.0 | 0.9 | 0.7 | 55 | 1.5065 | 2507.0544 |

### constant tuning

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **job order** | **booster** | **nfeatures** | **training times** | **scores (RMSE)** |
| 15 | constant | 1 | 1.2163 | 10381.0088 |

### gbtree tuning

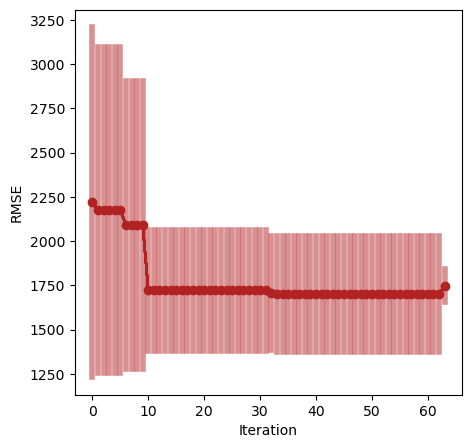
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **tree method** | **grow policy** | **max depth** | **max leaves** | **colsample bytree** | **subsample** | **nfeatures** | **training times** | **scores (RMSE)** |
| hist | depthwise | 8.0 | 256.0 | 0.6 | 0.9 | 44 | 6.7641 | 1723.1128 |
| hist | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 82 | 6.3002 | 1989.4866 |
| hist | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 82 | 6.3359 | 1989.4866 |
| hist | depthwise | 10.0 | 1024.0 | 0.8 | 0.7 | 37 | 4.4647 | 2089.4375 |

## Feature Evolution

During the Model and Feature Tuning Stage, Driverless AI evaluates the effects of different types of algorithms, algorithm parameters, and features. The goal of the Model and Feature Tuning Stage is to determine the best algorithm and parameters to use during the Feature Evolution Stage.

In the Feature Evolution Stage, Driverless AI trained constant, xgboost and lightgbm models (648) where each model evaluated a different set of features. The Feature Evolution Stage uses a genetic algorithm to search the large feature engineering space.

The graph below shows the effect the Model and Feature Tuning Stage and Feature Evolution Stage had on the performance.



Based on the experiment settings and column types in the dataset, Driverless AI was able to explore the following transformers:

* **BinnerTransformer**: None
* **InteractionsTransformer**: the Interactions Transformer adds, divides, multiplies, and subtracts two numeric columns in the data to create a new feature.
* **CatOriginalTransformer**: the Categorical Original Transformer applies an identity transformation that leaves categorical features as they are. This transformer works with models that can handle non-numeric feature values.
* **FrequentTransformer**: the Frequent Transformer calculates the frequency for each value in categorical column(s) and uses this as a new feature. This count can be either the raw count or the normalized count.
* **CatTransformer**: the Categorical Transformer sorts a categorical column in lexicographical order and uses the order index created as a new feature. This transformer works with models that can handle categorical features.
* **OneHotEncodingTransformer**: the One-hot Encoding transformer converts a categorical column to a series of boolean features by performing one-hot encoding. The boolean features are used as new features.
* **OneHotEncodingUnsupervisedTransformer**: None
* **IsHolidayTransformer**: the Is Holiday Transformer determines if a date column is a holiday. A boolean column indicating if the date is a holiday is added as a new feature. Creates a separate feature for holidays in the United States, United Kingdom, Germany, Mexico, and the European Central Bank. Other countries available in the python Holiday package can be added via the configuration file.
* **DateTimeDiffTransformer**: None
* **DatesTransformer**: the Date Transformer retrieves any date or time values, including: Year, Quarter, Month, Day, Day of Year, Week, Weekday, Hour, Minute, Second.
* **TextTransformer**: the Text Transformer tokenizes a text column and creates a TFIDF matrix (term frequency-inverse document frequency) or count (count of the word) matrix. This may be followed by dimensionality reduction using truncated SVD. Selected components of the TF-IDF/Count matrix are used as new features.
* **TextOriginalTransformer**: None
* **EwmaLagsTransformer**: the Exponentially Weighted Moving Average (EWMA) Transformer calculates the exponentially weighted moving average of target or feature lags.
* **LagsAggregatesTransformer**: the Lags Aggregates Transformer calculates aggregations of target/feature lags like mean(lag7, lag14, lag21) with support for mean, min, max, median, sum, skew, kurtosis, std. The aggregation is used as a new feature.
* **LagsTransformer**: the Lags Transformer creates target/feature lags, possibly over groups. Each lag is used as a new feature. Lag transformers may apply to categorical (strings) features or binary/multiclass string valued targets after they have been internally numerically encoded.
* **LagsInteractionTransformer**: the Lags Interaction Transformer creates target/feature lags and calculates interactions between the lags (lag2 - lag1, for instance). The interaction is used as a new feature.
* **MeanTargetTransformer**: None
* **TimeSeriesTargetEncTransformer**: None
* **ImageVectorizerTransformer**: the Image Vectorizer Transformer converts a column with an image path or URI to the embeddings representation. It utilizes ImageNet pre-trained models with a possible finetuning on the input data.
* **ImageOriginalTransformer**: None

**Dropped Features**

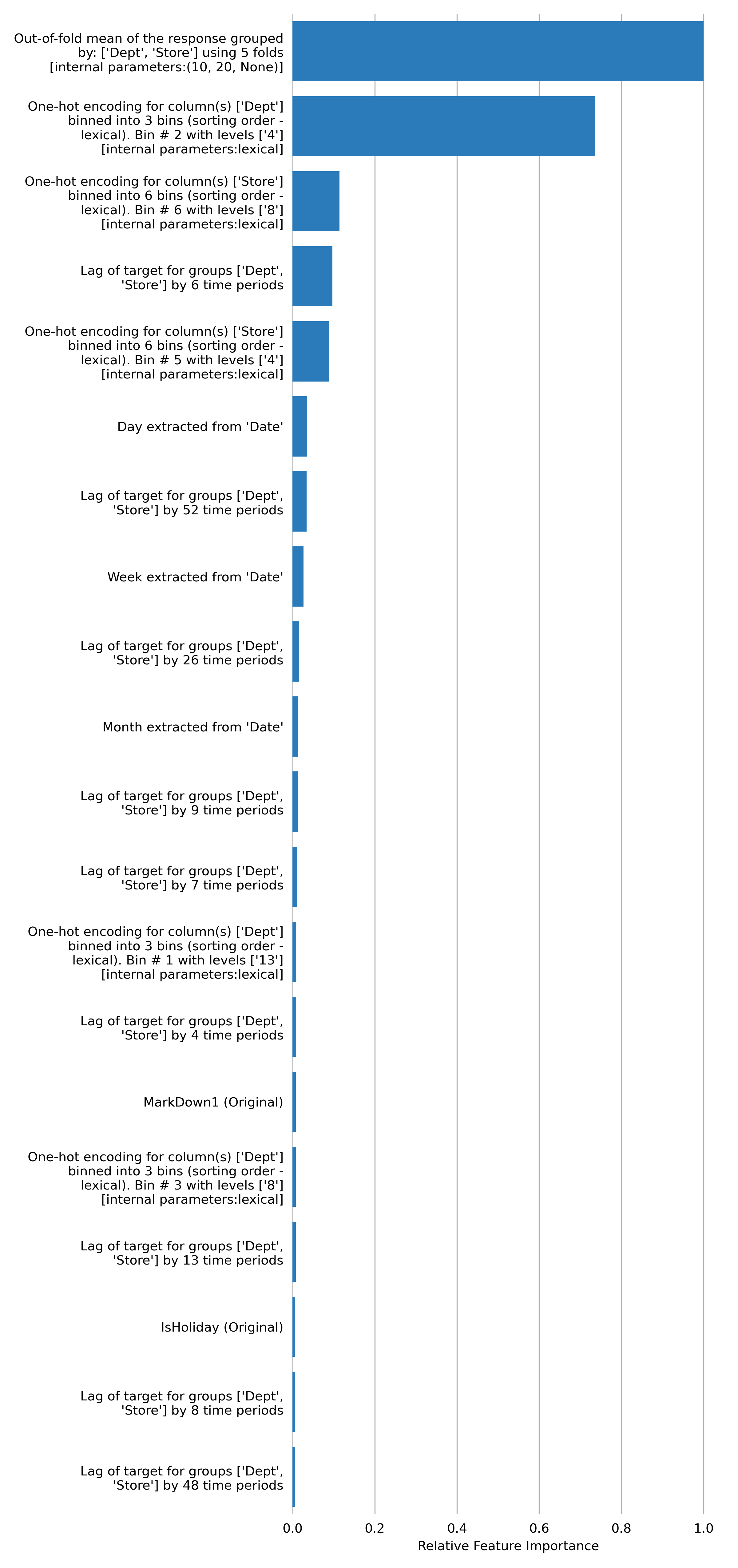
Below is the complete list of dropped features due to numerous reasons.

|  |  |
| --- | --- |
| **Name** | **Reason** |
| MarkDown2 | DAI Model Dropped |
| MarkDown3 | DAI Model Dropped |
| MarkDown4 | DAI Model Dropped |
| MarkDown5 | DAI Model Dropped |
| sample\_weight | User Dropped |

## Feature Transformations

The result of the Feature Evolution Stage is a set of features to use for the final model. Some of these features were automatically created by Driverless AI. The top features used in the final model are shown below, ordered by importance. The features in the table are limited to the top 50, restricted to those with relative importance greater than or equal to 0.003. If no transformer was applied, the feature is an original column.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Feature** | **Description** | **Transformer** | **Relative Importance** |
| 1 | 10\_CVTE: Dept: Store.0 | Out-of-fold mean of the response grouped by: ['Dept', 'Store'] using 5 folds [internal parameters:(10, 20, None)] | Cross Validation Target Encoding | 1.0 |
| 2 | 0\_OHE: Dept.2 | One-hot encoding for column(s) ['Dept'] binned into 3 bins (sorting order - lexical). Bin # 2 with levels ['4'] [internal parameters:lexical] | One Hot Encoding | 0.7358 |
| 3 | 1\_OHE: Store.6 | One-hot encoding for column(s) ['Store'] binned into 6 bins (sorting order - lexical). Bin # 6 with levels ['8'] [internal parameters:lexical] | One Hot Encoding | 0.1145 |
| 4 | 9\_TargetLag: Date: Dept: Store.6 | Lag of target for groups ['Dept', 'Store'] by 6 time periods | Lags | 0.0969 |
| 5 | 1\_OHE: Store.5 | One-hot encoding for column(s) ['Store'] binned into 6 bins (sorting order - lexical). Bin # 5 with levels ['4'] [internal parameters:lexical] | One Hot Encoding | 0.0887 |
| 6 | 8\_Date: Date~get\_day | Day extracted from 'Date' | Date Expansion | 0.0357 |
| 7 | 9\_TargetLag: Date: Dept: Store.52 | Lag of target for groups ['Dept', 'Store'] by 52 time periods | Lags | 0.0339 |
| 8 | 8\_Date: Date~get\_week | Week extracted from 'Date' | Date Expansion | 0.027 |
| 9 | 9\_TargetLag: Date: Dept: Store.26 | Lag of target for groups ['Dept', 'Store'] by 26 time periods | Lags | 0.0163 |
| 10 | 8\_Date: Date~get\_month | Month extracted from 'Date' | Date Expansion | 0.014 |
| 11 | 9\_TargetLag: Date: Dept: Store.9 | Lag of target for groups ['Dept', 'Store'] by 9 time periods | Lags | 0.0125 |
| 12 | 9\_TargetLag: Date: Dept: Store.7 | Lag of target for groups ['Dept', 'Store'] by 7 time periods | Lags | 0.011 |
| 13 | 0\_OHE: Dept.1 | One-hot encoding for column(s) ['Dept'] binned into 3 bins (sorting order - lexical). Bin # 1 with levels ['13'] [internal parameters:lexical] | One Hot Encoding | 0.0091 |
| 14 | 9\_TargetLag: Date: Dept: Store.4 | Lag of target for groups ['Dept', 'Store'] by 4 time periods | Lags | 0.009 |
| 15 | 3\_MarkDown1 | MarkDown1 (Original) | None | 0.0081 |
| 16 | 0\_OHE: Dept.3 | One-hot encoding for column(s) ['Dept'] binned into 3 bins (sorting order - lexical). Bin # 3 with levels ['8'] [internal parameters:lexical] | One Hot Encoding | 0.0081 |
| 17 | 9\_TargetLag: Date: Dept: Store.13 | Lag of target for groups ['Dept', 'Store'] by 13 time periods | Lags | 0.0078 |
| 18 | 2\_IsHoliday | IsHoliday (Original) | None | 0.0066 |
| 19 | 9\_TargetLag: Date: Dept: Store.8 | Lag of target for groups ['Dept', 'Store'] by 8 time periods | Lags | 0.0059 |
| 20 | 9\_TargetLag: Date: Dept: Store.48 | Lag of target for groups ['Dept', 'Store'] by 48 time periods | Lags | 0.0057 |
| 21 | 9\_TargetLag: Date: Dept: Store.5 | Lag of target for groups ['Dept', 'Store'] by 5 time periods | Lags | 0.0056 |
| 22 | 9\_TargetLag: Date: Dept: Store.11 | Lag of target for groups ['Dept', 'Store'] by 11 time periods | Lags | 0.0044 |
| 23 | 9\_TargetLag: Date: Dept: Store.15 | Lag of target for groups ['Dept', 'Store'] by 15 time periods | Lags | 0.004 |
| 24 | 9\_TargetLag: Date: Dept: Store.2 | Lag of target for groups ['Dept', 'Store'] by 2 time periods | Lags | 0.0037 |
| 25 | 9\_TargetLag: Date: Dept: Store.10 | Lag of target for groups ['Dept', 'Store'] by 10 time periods | Lags | 0.0034 |
| 26 | 1\_OHE: Store.1 | One-hot encoding for column(s) ['Store'] binned into 6 bins (sorting order - lexical). Bin # 1 with levels ['1'] [internal parameters:lexical] | One Hot Encoding | 0.0032 |
| 27 | 9\_TargetLag: Date: Dept: Store.3 | Lag of target for groups ['Dept', 'Store'] by 3 time periods | Lags | 0.0031 |



**Monotonicity Constraints Details**

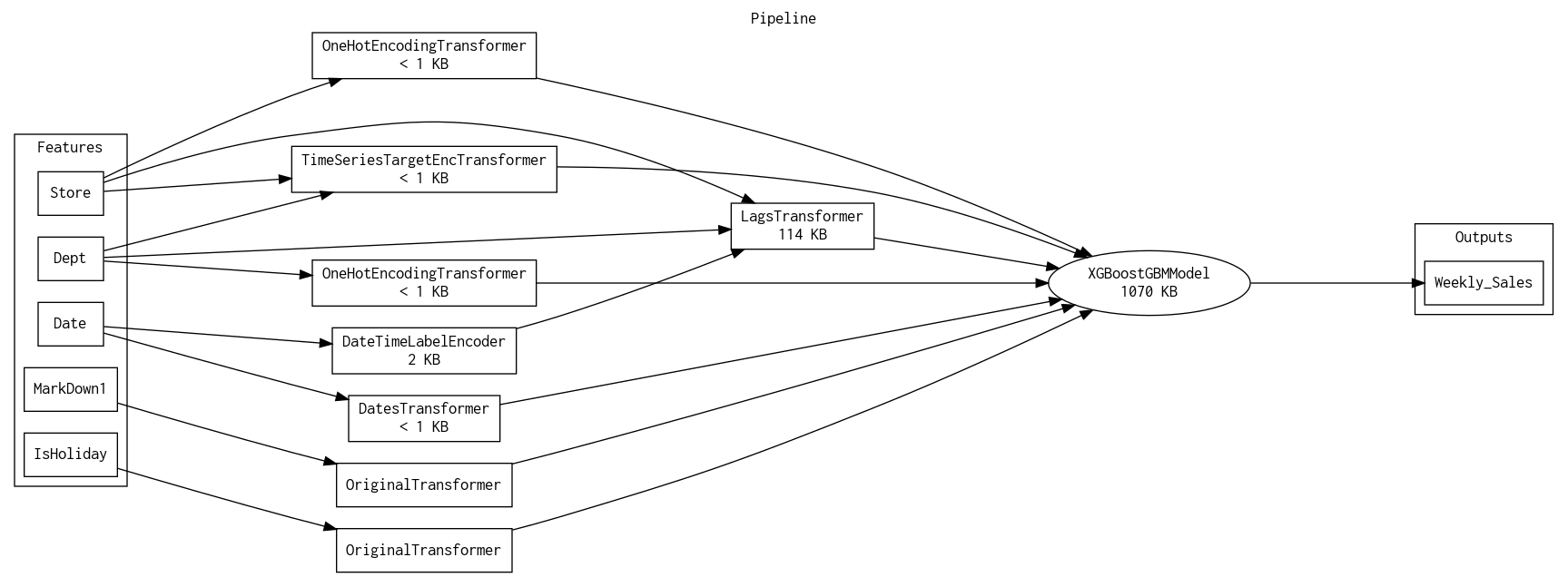
Monotonicity constraints enforce a monotonic relationship between a specified feature and the target prediction. For example, given a model trained to predict housing prices, we might want to enforce that the model predicts higher housing prices with increasing lot size and lower housing prices with increasing neighborhood crime rate.

This experiment enables automatic monotonicity constraints, which mean Driverless AI automatically determines if monotonicity is present and then enforces it through all or part of the modeling pipelines. Depending on the level of feature-target correlation, Driverless AI assigns positive, negative, or no monotonicity constraints. Specifically, monotonicity constraints are enforced if the absolute correlation is greater than 0.1 (the monotonicity correlation threshold set for this experiment). Driverless AI does not enforce monotonicity constraints for features below this correlation threshold.

## Final Model

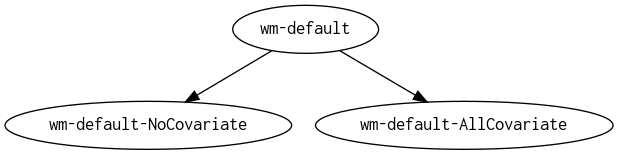
**Pipeline**

Final XGBoostGBMModel pipeline with ensemble\_level=0 transforming 5 original features -> 40 features in each of 1 models each fit on time-based hold-out.:



**Model Lineage**

The following plot shows the experiment lineage for the current experiment wm-default-AllCovariate.



**Details**

* The fitted features of the final model are the best features found during the feature engineering iterations.
* The target transformer indicates the type of transformation applied to the target column.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Index** | **Type** | **Model Weight** | **Fitted features** | **Target Transformer** |
| 0 | XGBoostGBMModel | 1 | 40 | identity\_noclip |

* Model Index: 0 has a weight of 1 in the final ensemble

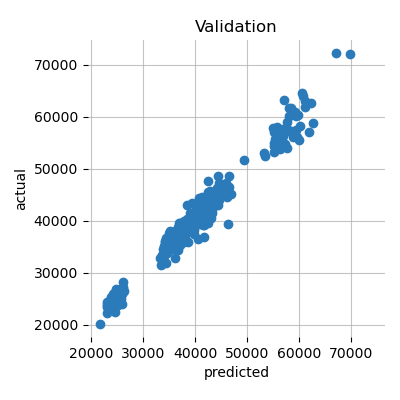
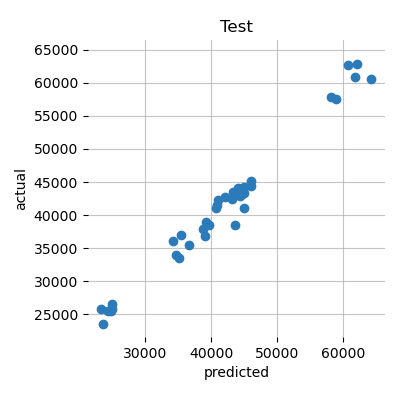
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Type** | **subsample** | **grow policy** | **colsample bytree** | **tree method** | **max leaves** | **index** | **Split Type** | **model class name** | **max depth** | **learning rate** |
| XGBoostGBMModel | 0.9 | lossguide | 0.6 | hist | 256 | 0 | None | XGBoostGBMModel | 8 | 0.05 |

For a complete list of the parameters of the final model, see the Appendix.

**Performance of Final Model**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Scorer** | **Optimized** | **Better score is** | **Final ensemble scores on validation (internal or external holdout(s)) data** | **Final ensemble standard deviation on validation (internal or external holdout(s)) data** | **Final test scores** | **Final test standard deviation** |
| RMSE | \* | lower | 1747.488 | 93.58347 | 1706.288 | 244.6643 |
| GINI |  | higher | 0.9796952 | 0.002492234 | 0.9828831 | 0.01064928 |
| MAE |  | lower | 1316.992 | 69.32006 | 1321.424 | 173.1133 |
| MAPE |  | lower | 3.243409 | 0.147012 | 3.37104 | 0.4296199 |
| MER |  | lower | 2.702249 | 0.2008379 | 2.91315 | 0.493346 |
| MSE |  | lower | 3053715 | 321807.2 | 2911419 | 835739.2 |
| R2COD |  | higher | 0.9717062 | 0.003166344 | 0.9739551 | 0.01254608 |
| R2 |  | higher | 0.9724222 | 0.003111068 | 0.978713 | 0.007376826 |
| RMSLE |  | lower | 0.04159058 | 0.001801287 | 0.04308536 | 0.005444971 |
| RMSPE |  | lower | 4.157137 | 0.1905621 | 4.382938 | 0.5971275 |
| SMAPE |  | lower | 3.254463 | 0.1475065 | 3.340506 | 0.4084651 |

*Actual vs Predicted*

## Alternative Models

During the experiment, Driverless AI trained 182 alternative models. The following algorithms were evaluated during the Driverless AI experiment:

|  |  |  |  |
| --- | --- | --- | --- |
| **algorithm** | **package** | **version** | **documentation** |
| lightgbm | lightgbm | 3.3.2.99 | LightGBM, Light Gradient Boosting Machine. Contributors: https://github.com/microsoft/LightGBM/graphs/contributors. |
| constant | custom package | 1.10.7.2 | reference model that predicts a constant aimed at minimizing the given scorer |
| gbtree | xgboost | 1.5.0-dev | XGBoost: eXtreme Gradient Boosting library. Contributors: https://github.com/dmlc/xgboost/blob/master/CONTRIBUTORS.md |

Driverless AI can evaluate an array of algorithms, including but not limited to XGBoost GBM, XGBoost Dart, XGBoost GLM, LightGBM, RuleFit, Tensorflow, and FTRL models. The table below explains why certain algorithms were not selected for the final model, if any.

|  |  |
| --- | --- |
| **algorithm** | **selection** |
| gblinear | algorithm not evaluated due to experiment configuration |
| decision tree | algorithm not evaluated due to experiment configuration |
| rulefit | algorithm not evaluated due to experiment configuration |
| tensorflow | algorithm not evaluated due to experiment configuration |
| ftrl | algorithm not evaluated due to experiment configuration |
| dart | algorithm not evaluated due to experiment configuration |
| lightgbm | not selected due to low performance during feature evolution stage |
| gbtree | selected for final model |

## Deployment

For this experiment, both Python and MOJO Scoring Pipelines are available for productionizing the final model pipeline for a given row of data or table of data.

### Python Scoring Pipeline

This package contains an exported model and Python 3.6 source code examples for productionizing models built using H2O Driverless AI. The Python Scoring Pipeline is located here:

* **yuki.shimada@h2o.ai/h2oai\_experiment\_cf326492-f3f4-11ef-a69c-4e030d02e593/scoring\_pipeline/scorer.zip**

The files in this package allow you to transform and score on new data in a couple of different ways:

* From Python 3.6, you can import a scoring module, then use the module to transform and score on new data.
* From other languages and platforms, you can use the TCP/HTTP scoring service bundled with this package to call into the scoring pipeline module through remote procedure calls (RPC).

### MOJO Scoring Pipeline

Note: The MOJO Scoring Pipeline is currently in a beta state. Updates and improvements will continue to be made in subsequent Driverless AI releases. The MOJO Scoring Pipeline is located here:

* **yuki.shimada@h2o.ai/h2oai\_experiment\_cf326492-f3f4-11ef-a69c-4e030d02e593/mojo\_pipeline/mojo.zip**

For completed experiments, Driverless AI converts models to MOJOs (Model Objects, Optimized). A MOJO is a scoring engine that can be deployed in any Java environment for scoring in real time.

## Partial Dependence Plots

Partial dependence plots show the partial dependence as a function of specific values for a feature subset. The plots show how machine-learned response functions change based on the values of an input feature of interest, while taking nonlinearity into consideration and averaging out the effects of all other input features. Partial dependence plots enable increased transparency in a model and enable the ability to validate and debug a model by comparing a feature’s average predictions across its domain to known standards and reasonable expectations.

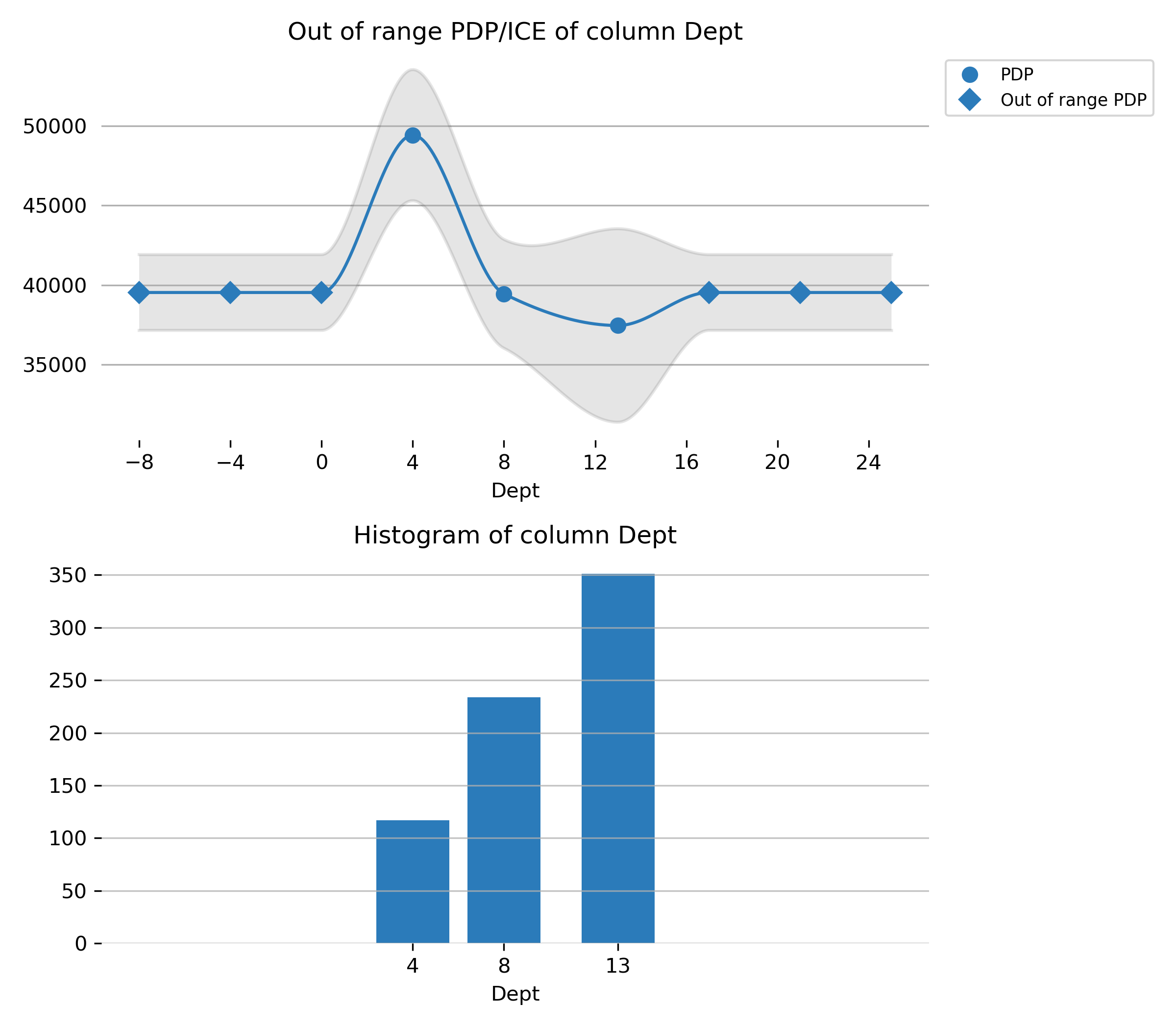
The partial dependence plots are shown for the top 5 original variables. The top 5 original variables are chosen based on their Component Based Variable Importance.

**Plot Details**

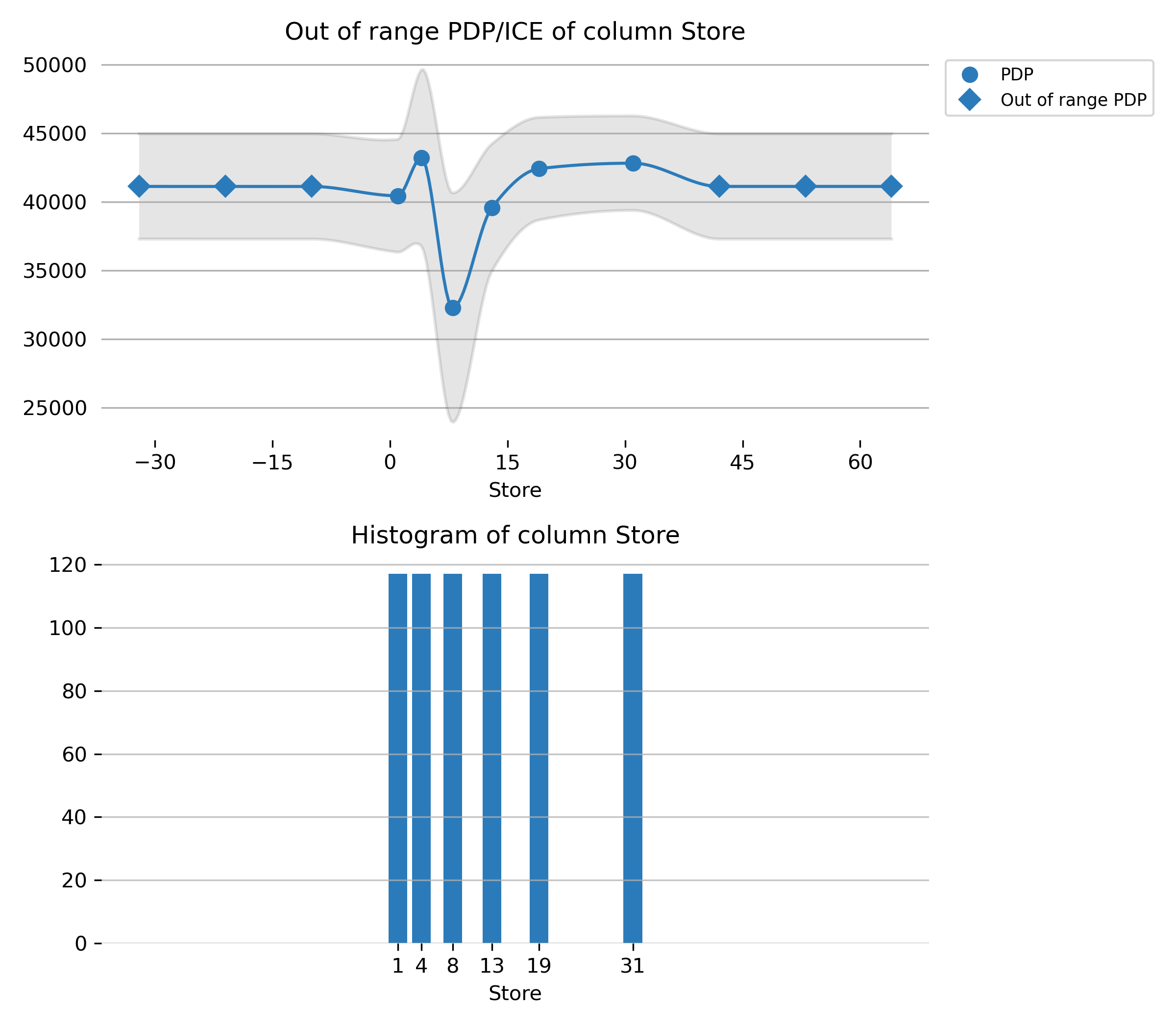
In the Driverless AI PDP, the y-axis represents the mean response, and a shaded region (for numeric features) or shaded bar (for categorical features) represents ± 1 standard deviation. Out-of-range PDP (diamond markers) represent values outside feature intervals seen in the data, unseen categorical values, or missing values.

For continuous features, numeric values up to 3 standard deviations lower than the minimum training value and higher than the maximum training value are feed into the model. For categorical features, an unseen categorical value is feed into the model denoted by UNSEEN (if the categorical value “UNSEEN” already exists in the training data, the out-of-range is done on a value called “UNSEEN\_[x],” where x is some integer).

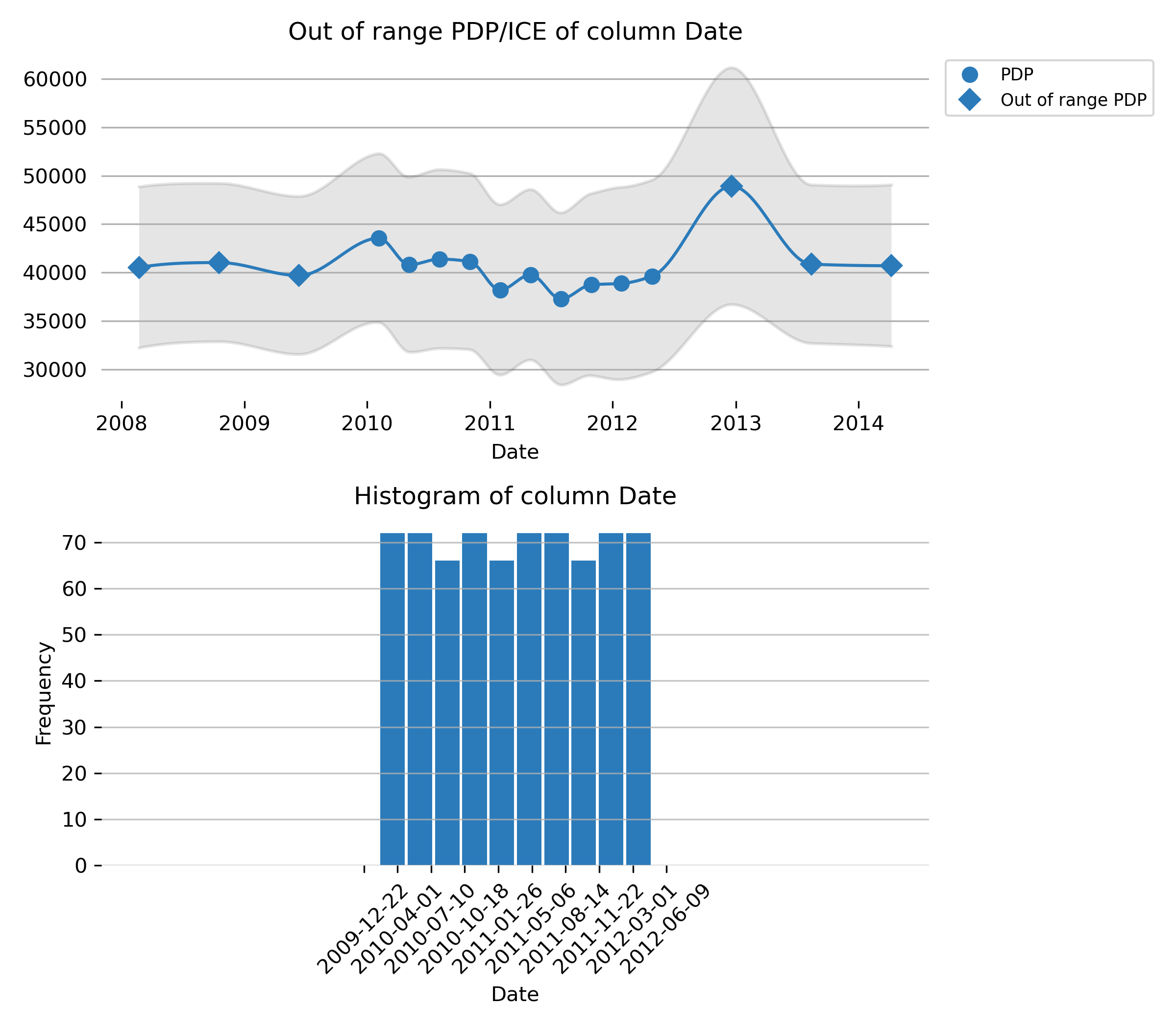
Feature **Dept**



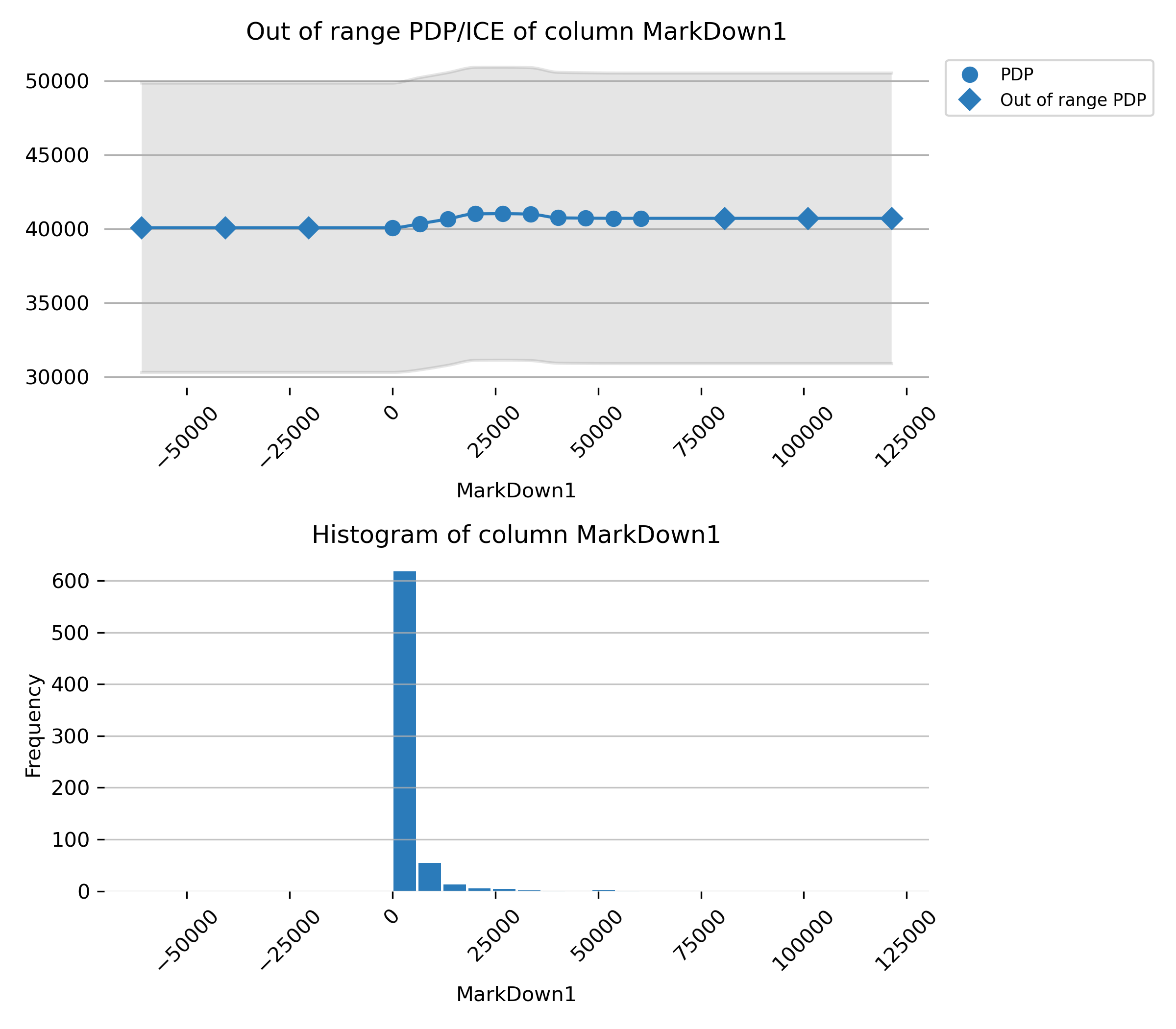
Feature **Store**



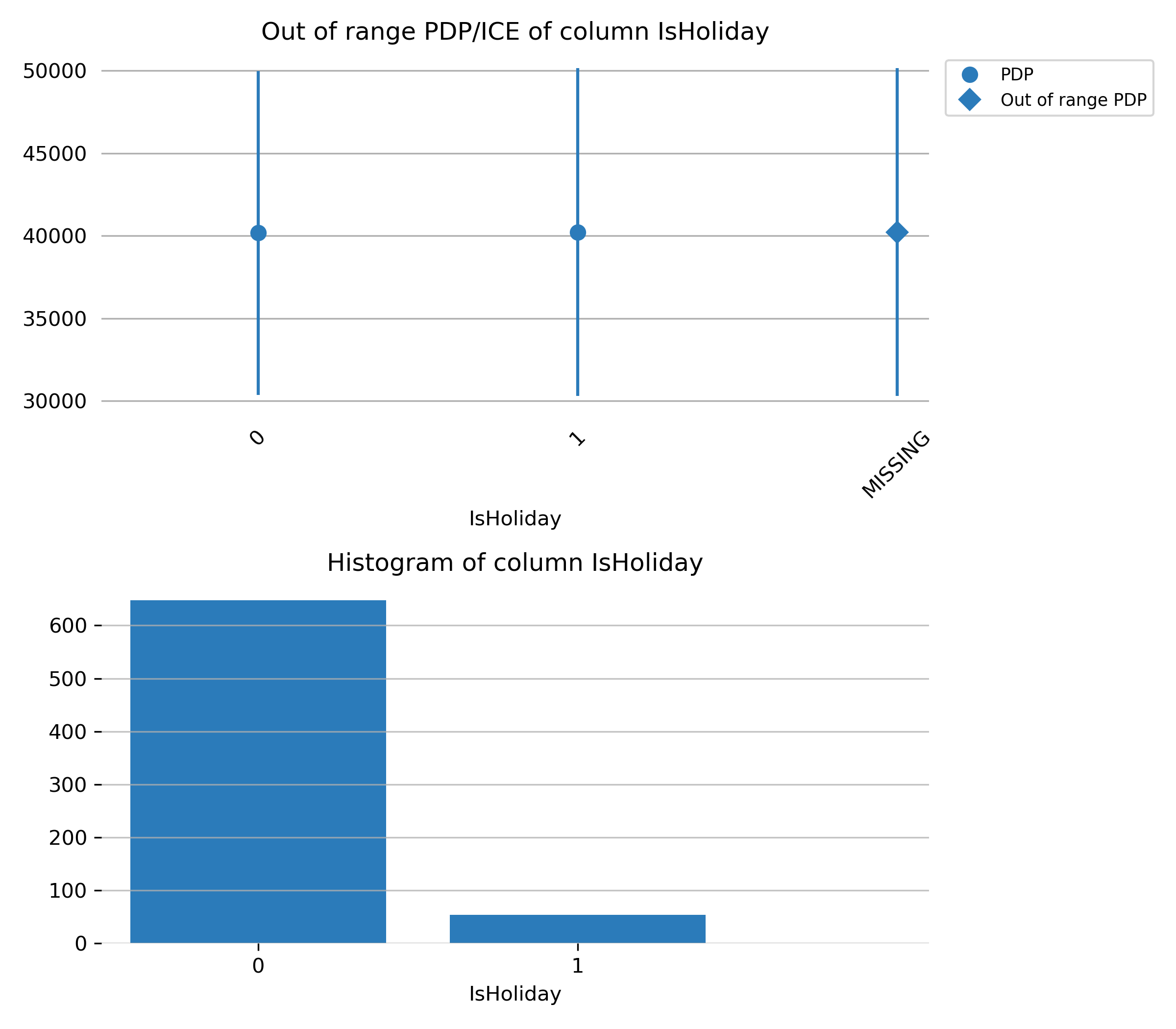
Feature **Date**



Feature **MarkDown1**



Feature **IsHoliday**

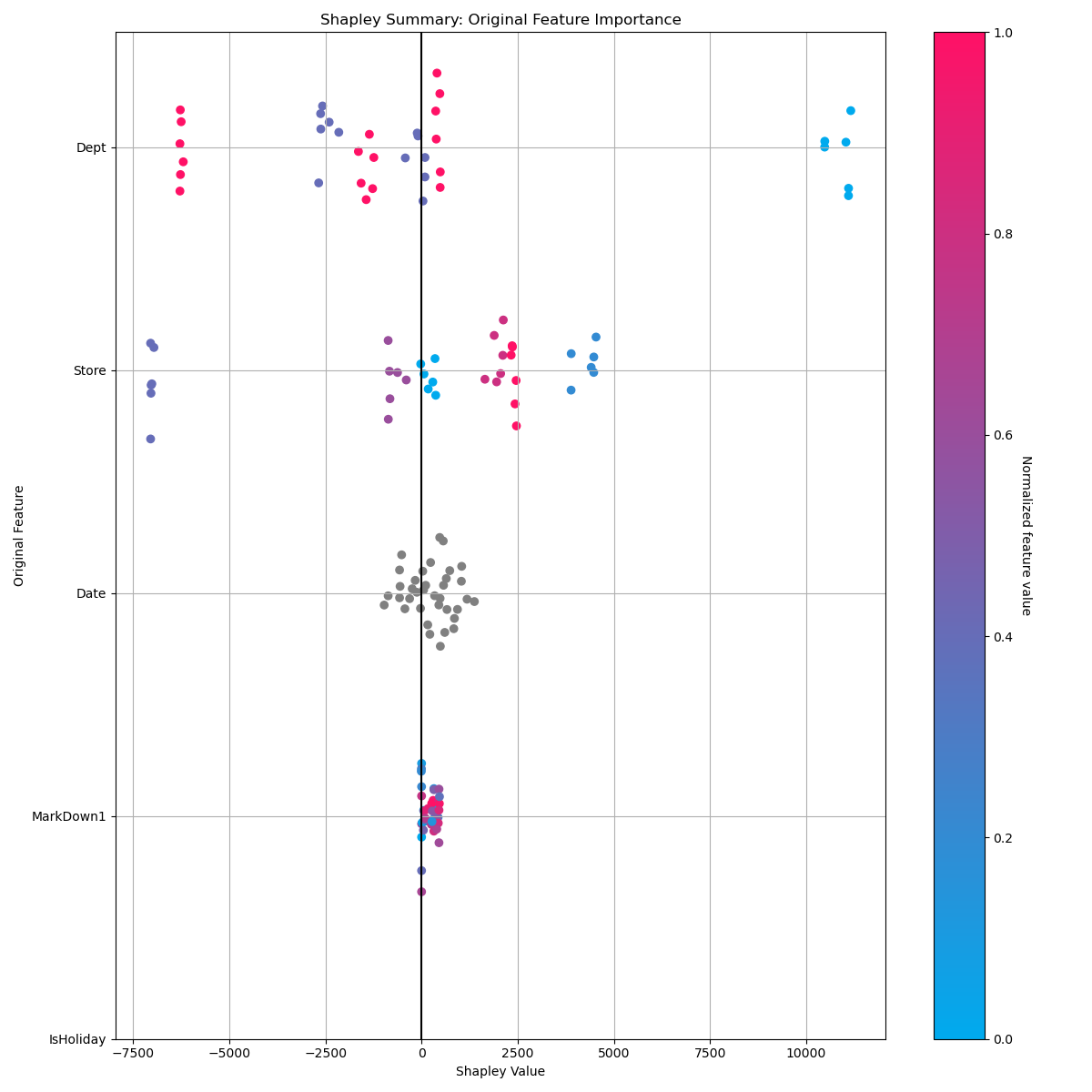


## Shapley Values

#### Shapley Contributions on the Test Dataset

Shapley explanations are a technique with credible theoretical support that presents consistent global and local feature contributions. For regression problems, local Shapley feature contributions plus the bias term sum to the final model’s prediction. For classification problems, they sum to the prediction before applying the link function.

This section uses Driverless AI’s Naive Shapley method to calculate local Shapley explanations for original features. These explanations are approximation for the original features and are based on how often the features are used in transformed features, and how important those transformed features are to the final model. The importance of each transformed feature is distributed equally to all original features that helped create it. This is then summed for each original feature.



## Appendix

### Final Model Details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Index** | **Type** | **Model Weight** | **Fitted features** | **Target Transformer** |
| 0 | XGBoostGBMModel | 1 | 40 | identity\_noclip |

**Model Index: Final Model - Single Model**

|  |  |
| --- | --- |
| **parameter** | **value** |
| base\_score | 39868.8125 |
| booster | gbtree |
| colsample\_bytree | 0.6 |
| debug\_verbose | 0 |
| disable\_gpus | True |
| eval\_metric | mae |
| gamma | 0.01 |
| grow\_policy | lossguide |
| imbalance\_info\_dict | {} |
| importance\_type | gain |
| label\_counts |  |
| labels | None |
| learning\_rate | 0.05 |
| max\_bin | 128 |
| max\_delta\_step | 0.0 |
| max\_depth | 8 |
| max\_leaves | 256 |
| min\_child\_weight | 2 |
| model\_class\_name | XGBoostGBMModel |
| model\_id | Final Model - Single Model |
| monotone\_constraints | (-1,1,0,1,0,-1,1,1,1,-1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1) |
| monotonicity\_constraints | True |
| n\_estimators | 372 |
| n\_gpus | 0 |
| n\_jobs | 3 |
| nthread | 3 |
| num\_class | 1 |
| objective | reg:squarederror |
| reg\_alpha | 0.1 |
| reg\_lambda | 5.0 |
| scale\_pos\_weight | 1.0 |
| score\_f\_name | RMSE |
| seed | 299670398 |
| silent | 1 |
| subsample | 0.9 |
| tree\_method | hist |
| nfolds | 1 |

### Config Overrides

The Config Overrides represent the fine-control parameters.

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| prob\_lagsinteraction | 0.2 |
| auth\_openid\_token\_introspection\_url | http://localhost:9876/introspect |
| auth\_oidc\_introspection\_endpoint\_url | https://auth.internal.dedicated.h2o.ai/auth/realms/hac/protocol/openid-connect/token/introspect |
| http\_cookie\_attributes | {'path': '/workspaces/default/daiEngines/dai'} |
| memory\_limit\_gb | 0 |
| tournament\_remove\_poor\_scores\_before\_final\_model\_factor | 1.1 |
| disablelogout | True |
| num\_gpus\_per\_experiment | 0 |
| base\_url | /workspaces/default/daiEngines/dai/ |
| skip\_record\_h2o4gpu\_version | True |
| datetime\_funcs | ['day', 'month', 'quarter', 'week', 'year'] |
| threshold\_scorer | F1 |
| override\_virtual\_cores | 6 |
| auth\_oidc\_issuer\_url | https://auth.internal.dedicated.h2o.ai/auth/realms/hac |
| server\_ip | 10.3.2.136 |
| prob\_lagsaggregates | 0.2 |
| stabilize\_fs | False |
| auth\_openid\_scope | openid offline\_access profile ai.h2o.storage |
| num\_gpus\_per\_model | 0 |
| tournament\_keep\_absolute\_ok\_scores\_before\_evolution\_model\_factor | 1.1 |
| h2o\_storage\_tls\_ca\_path | /pki/ca/ca-cert |
| hac\_link\_url | https://internal.dedicated.h2o.ai |
| skip\_track\_system\_info\_startup | True |
| cv\_in\_cv\_overconfidence\_protection | on |
| one\_hot\_encoding\_cardinality\_threshold | 6 |
| included\_models | ['Aggregator', 'Constant', 'DecisionTree', 'FTRL', 'GLM', 'ImageAuto', 'ImbalancedLightGBM', 'ImbalancedXGBoostGBM', 'IsolationForestAnomaly', 'KMeans', 'KMeansFreq', 'KMeansOHE', 'LightGBM', 'LightGBMDask', 'RuleFit', 'TensorFlow', 'TextALBERT', 'TextBERT', 'TextCamemBERT', 'TextDistilBERT', 'TextMultilingualBERT', 'TextRoBERTa', 'TextXLM', 'TextXLMRoberta', 'TextXLNET', 'TorchGrowNet', 'TruncSVD', 'Unsupervised', 'UnsupervisedGeneral', 'UnsupervisedOneAll', 'UnsupervisedOneOutput', 'XGBoostDart', 'XGBoostDartDask', 'XGBoostGBM', 'XGBoostGBMDask', 'XGBoostRF', 'XGBoostRFDask', 'ZeroInflatedLightGBM', 'ZeroInflatedXGBoost'] |
| api\_token\_introspection\_enabled | True |
| h2o\_storage\_tls\_key\_path | /pki/client/client-key |
| prob\_lag\_non\_targets | 0.1 |
| enable\_xgboost\_h2o\_recipes | False |
| auth\_oidc\_post\_logout\_url | https://enginemanager.internal.dedicated.h2o.ai/workspaces/default/daiEngines/dai/oidc/login |
| enable\_startup\_checks | False |
| h2o\_mlops\_ui\_url | https://mlops.internal.dedicated.h2o.ai |
| auth\_oidc\_username\_claim | preferred\_username |
| h2o\_storage\_projects\_enabled | True |
| auth\_openid\_redirect\_uri | https://enginemanager.internal.dedicated.h2o.ai/workspaces/default/daiEngines/dai/oidc/callback |
| stabilize\_varimp | False |
| application\_id | dai\_7 |
| tournament\_remove\_worse\_than\_constant\_before\_evolution | False |
| h2o\_drive\_endpoint\_url | http://h2o-drive-userdrive-compat.drive:7112 |
| cuda\_visible\_devices |  |
| h2o\_storage\_tls\_cert\_path | /pki/client/client-cert |
| auth\_oidc\_identity\_source | id\_token |
| auth\_openid\_use\_objectpath\_match | True |
| enable\_quick\_benchmark | False |
| worker\_ip | 10.3.2.136 |
| fast\_startup | True |
| included\_scorers | ['ACCURACY', 'AUC', 'AUCPR', 'CALINSKIHARABASZ', 'DAVIESBOULDIN', 'F05', 'F1', 'F2', 'FDR', 'FNR', 'FOR', 'FPR', 'GINI', 'LOGLOSS', 'MACROAUC', 'MACROF1', 'MACROMCC', 'MAE', 'MAPE', 'MCC', 'MER', 'MSE', 'NPV', 'PRECISION', 'R2', 'R2COD', 'RECALL', 'RMSE', 'RMSLE', 'RMSPE', 'SILHOUETTE', 'SMAPE', 'TNR', 'UNSUPERVISED'] |
| tournament\_remove\_poor\_scores\_before\_evolution\_model\_factor | 1.1 |
| auth\_openid\_use\_objectpath\_expression | $.sub is "bc34ef4d-ff29-4375-8bab-04d860fcac32" |
| last\_exclusive\_mode | safe |
| auth\_oidc\_token\_endpoint\_url | https://auth.internal.dedicated.h2o.ai/auth/realms/hac/protocol/openid-connect/token |
| authentication\_method | oidc |
| ohe\_bin\_list | [10, 50, 100] |
| check\_system | False |
| max\_runtime\_minutes | 60 |
| how\_started | GUI |
| last\_recipe | auto |
| included\_transformers | ['AggregatorTransformer', 'AutovizRecommendationsTransformer', 'BERTTransformer', 'BinnerTransformer', 'CVCatNumEncodeTransformer', 'CVTargetEncodeTransformer', 'CatOriginalTransformer', 'CatTransformer', 'ClusterDistTransformer', 'ClusterIdAllNumTransformer', 'ClusterTETransformer', 'DateOriginalTransformer', 'DateTimeDiffTransformer', 'DateTimeOriginalTransformer', 'DatesTransformer', 'EwmaLagsTransformer', 'FrequentTransformer', 'ImageOriginalTransformer', 'ImageVectorizerTransformer', 'InteractionsTransformer', 'IsHolidayTransformer', 'IsolationForestAnomalyAllNumericTransformer', 'IsolationForestAnomalyNumCatAllColsTransformer', 'IsolationForestAnomalyNumCatTransformer', 'IsolationForestAnomalyNumericTransformer', 'LagsAggregatesTransformer', 'LagsInteractionTransformer', 'LagsTransformer', 'LexiLabelEncoderTransformer', 'MeanTargetTransformer', 'NumCatTETransformer', 'NumToCatTETransformer', 'NumToCatWoEMonotonicTransformer', 'NumToCatWoETransformer', 'OneHotEncodingTransformer', 'OneHotEncodingUnsupervisedTransformer', 'OriginalTransformer', 'RobustScalerTransformer', 'StandardScalerTransformer', 'StringConcatTransformer', 'TextBiGRUTransformer', 'TextCNNTransformer', 'TextCharCNNTransformer', 'TextLinModelTransformer', 'TextOriginalTransformer', 'TextTransformer', 'TimeSeriesTargetEncSimpleTransformer', 'TimeSeriesTargetEncTransformer', 'TruncSVDAllNumTransformer', 'TruncSVDNumTransformer', 'WeightOfEvidenceTransformer'] |
| prob\_default\_lags | 0.2 |
| h2o\_storage\_address | mlops-storage.mlops:9999 |
| disk\_limit\_gb | 0 |
| virtual\_cores\_per\_physical\_core | 2 |