

# Project 1: predictions of the emissions and electricity consumption

Denis Storozhuk, Insy2s

Data link: <https://data.seattle.gov/dataset/2016-Building-Energy-Benchmarking/2bpz-gwpy>

# Agenda

---

- 0. Brief and goals setting
- 1. Python environment:
  - Data cleaning and transformation
  - Choosing the best model
  - Hyperparameter tuning
- 2. BigQuery environment:
  - BQ ML: ETL data, modelling
  - Vertex AI Auto ML: ETL, modelling
- 3. Summary and key learnings

# Brief and goals setting

---

- Since our goal is to predict emissions ( $Y_0$ ) and electricity consumption ( $Y_1$ ), all the variables of electricity consumption to be deleted from  $X$
- $Y$  emissions - TotalGHGEmissions
- $Y$  electricity - Electricity(kWh)
- Regression problem

# Python Environment

# Load the dataset

---

- 1/ load the latest data from the link, draft data analysis
  - 2/ changing coma in the false object columns and transforming them to numerical (a problem appears while using Jupiter Cloud version)
- dropping anomalies - below 0 values for Y columns (TotalGHGEmissions, Electricity(kWh))

# Delete non-informative columns

---

3/ deleting Id and non-informative columns for modelling,

Columns to delete:

OSEBuildingID, DataYear, PropertyName, Address, City, State, TaxParcelIdentificationNumber,  
YearsENERGYSTARCertified, DefaultData, Comments, Outlier

dropping the columns with electricity and gas consumption data and GHGEmissionsIntensity

'SiteEUI(kBtu/sf)', 'SiteEUIWN(kBtu/sf)', 'SourceEUI(kBtu/sf)', 'SourceEUIWN(kBtu/sf)', 'SiteEnergyUse(kBtu)', 'SiteEnergyUseWN(kBtu)', 'SteamUse(kBtu)', 'Electricity(kWh)', 'Electricity(kBtu)', 'NaturalGas(therms)', 'NaturalGas(kBtu)', 'GHGEmissionsIntensity'

# Categorical and numerical data transformation

---

4/ splitting data into numerical and categorical

5/ replacing nans by median/most frequent

6/ implementing automatic Anova approach for treatment of the categorical columns, delete the columns which have no impact on Y (TotalGHGEmissions) (with  $p\_value > 0.05$ )

7/ selecting numerical columns with weak correlation (threshold  $< 0.5$ ) to be deleted

8/ implementing OrdinalEncoder for the categorical data

9/ perturbation of the dataset (PS: avoided due to small size of the dataset and problems with reproducibility)

# Modelling and hyperparameter tuning

---

- 10/ outliers removal (therefore no need to keep the Outlier column which was dropped initially)
- 11/ scaling/standardizing data (Scaler)
- 14/ splitting data into X and y (TotalGHGEmissions), yel (Electricity(kWh))
- 15/ train\_test\_split of the dataset (check random\_state to be equal for the two splits)
- 16/ checking performance of a set of the different models with fit/predict (KFold/cross\_val\_score can't be used due to a small size of a dataset and problems with reproducibility)
- 17/ retrieving features importance
- 18/ hyperparameter tuning to improve the model performance

# Final list of the columns

---

**Final list of X columns:** BuildingType, PrimaryPropertyType, Neighborhood, PropertyGFATotal, PropertyGFABuilding(s), ListOfAllPropertyUseTypes, LargestPropertyUseType, LargestPropertyUseTypeGFA, SecondLargestPropertyUseType, ThirdLargestPropertyUseType, NaturalGas(therms)

**Y columns:** TotalGHGEmissions, Electricity(kWh)

Regression problem

# Summary of the results for test data

---

<b>Model / Y</b>	<b>R2</b>	<b>RMSE</b>
Python: Random Forest (TotalGHGEmissions)	0.89	62.3
Python: XGB (Electricity(kWh))	0.88	950 769

# BigQuery ML

# ETL data

---

- Though BQ ML automatically treats nans, I encountered problems with categorical columns (String data type in BQ). The models showed bad performance
- Therefore I did manual data transformation with `ML.LABEL_ENCODER` for categorical columns (String data type in BQ) for BQ ML
- Additional data transformation (optional since it didn't improve the results) – replace nans with median/most frequent (`PERCENTILE_CONT`, `APPROX_TOP_COUNT`)

# Manual encoding for string columns

Type to search ?

Viewing workspace resources. SHOW STARRED ONLY

- dataset
- Models (10)
  - electricity\_bt\_mo...
  - electricity\_dnn\_m...
  - electricity\_lr\_model
  - electricity\_rf\_model
  - emission\_bt\_model
  - emission\_bt\_mod...
  - emission\_bt\_mod...
  - emission\_dnn\_mo...
  - emission\_lr\_model
  - emission\_rf\_model
- table\_all\_cols
- table\_clean
- ...

emission\_dnn\_model QUERY MODEL DELETE MODEL EXPORT MODEL REFRESH

DETAILS TRAINING EVALUATION INTERPRETABILITY SCHEMA

## Features

Filter Enter property name or value ?

Field name	Type	Mode	Description
<a href="#">BuildingType</a>	INT64	NULLABLE	
<a href="#">LargestPropertyUseType</a>	INT64	NULLABLE	
<a href="#">LargestPropertyUseTypeGFA</a>	INT64	NULLABLE	
<a href="#">ListOfAllPropertyUseTypes</a>	INT64	NULLABLE	
<a href="#">NaturalGas_therms_</a>	FLOAT64	NULLABLE	
<a href="#">Neighborhood</a>	INT64	NULLABLE	
<a href="#">PrimaryPropertyType</a>	INT64	NULLABLE	
<a href="#">PropertyGFABuilding_s_</a>	INT64	NULLABLE	
<a href="#">PropertyGFATotal</a>	INT64	NULLABLE	
<a href="#">SecondLargestPropertyUseType</a>	INT64	NULLABLE	
<a href="#">ThirdLargestPropertyUseType</a>	INT64	NULLABLE	

Rows per page: 50 ▾ 1 – 11 of 11 < >

# DNN model training (TotalGHGEmissions)

Type to search

Viewing workspace resources.

SHOW STARRED ONLY

- dataset
- Models (10)
  - electricity\_bt\_mo...
  - electricity\_dnn\_m...
  - electricity\_lr\_model
  - electricity\_rf\_model
  - emission\_bt\_model
  - emission\_bt\_mod...
  - emission\_bt\_mod...
  - emission\_dnn\_mo...
  - emission\_lr\_model
  - emission\_rf\_model
- table\_all\_cols

emission\_dnn\_model

QUERY MODEL DELETE MODEL EXPORT MODEL REFRESH

DETAILS TRAINING EVALUATION INTERPRETABILITY SCHEMA

### Loss

Iteration	Training loss	Evaluation loss
0	~250,000	~250,000
2	~10,000	~60,000
4	~10,000	~20,000
6	~10,000	~40,000
8	~10,000	~20,000
10	~10,000	~20,000
12	~10,000	~20,000

# Evaluation and FI (TotalGHGEmissions)

?

Viewing workspace resources.

[SHOW STARRED ONLY](#)

- dataset
- Models (10)
  - electricity\_bt\_mo...
  - electricity\_dnn\_m...
  - electricity\_lr\_model
  - electricity\_rf\_model
  - emission\_bt\_model
  - emission\_bt\_mod...
  - emission\_bt\_mod...
  - emission\_dnn\_mo... selected
  - emission\_lr\_model
  - emission\_rf\_model
- table\_all\_cols
- table\_clean
- table\_clean\_test
- table\_clean\_test\_num

## emission\_dnn\_model

[QUERY MODEL](#) [DELETE MODEL](#) [EXPORT MODEL](#) [REFRESH](#)

[DETAILS](#) [TRAINING](#) [EVALUATION](#) [INTERPRETABILITY](#) [SCHEMA](#)

### Explainable AI

Explainable AI provides a way of explaining the model and the predictions it produces. The following contains the features with the largest importance scores for your model overall. For explainable prediction on a per example basis, please run ML.EXPLAIN\_PREDICT or ML.EXPLAIN\_FORECAST.

Feature Name	Attribution
NaturalGas_therms_	47.402
LargestPropertyUseTypeGFA	18.672
ThirdLargestPropertyUseType	18.157
ListOfAllPropertyUseTypes	15.494
LargestPropertyUseType	11.543
PrimaryPropertyType	10.408
PropertyGFABuilding_s_	8.908
PropertyGFATotal	6.788
SecondLargestPropertyUseType	5.962
BuildingType	4.781
Neighborhood	3.648

## emission\_dnn\_model

[QUERY MODEL](#)

[DETAILS](#) [TRAINING](#) [EVALUATION](#)

Mean absolute error	56.9436
Mean squared error	25,423.1228
Mean squared log error	0.9273
Median absolute error	21.0876
R squared	0.9438

[PERSONAL HISTORY](#) [PROJECT HISTORY](#) [REFRESH](#) ^

# Evaluation and FI (Electricity(kWh))

Type to search ?

Viewing workspace resources.

[SHOW STARRED ONLY](#)

- feature\_engineering\_mode... ⋮
- External connections ⋮
- dataset ⋮
- Models (10) ⋮
  - electricity\_bt\_mo... ☆ ⋮
  - electricity\_dnn\_m... ☆ ⋮
  - electricity\_lr\_model ☆ ⋮
  - electricity\_rf\_model** ☆ ⋮
  - emission\_bt\_model ☆ ⋮
  - emission\_bt\_mod... ☆ ⋮
  - emission\_bt\_mod... ☆ ⋮
  - emission\_dnn\_mo... ☆ ⋮
  - emission\_lr\_model ☆ ⋮
  - emission\_rf\_model ☆ ⋮
- table\_all\_cols ☆ ⋮

## electricity\_rf\_model

[QUERY MODEL](#) [DELETE MODEL](#) [EXPORT MODEL](#) [REFRESH](#)

[DETAILS](#) [TRAINING](#) [EVALUATION](#) [INTERPRETABILITY](#) [SCHEMA](#)

### Explainable AI

Explainable AI provides a way of explaining the model and the predictions it produces. The following contains the features with the largest importance scores for your model overall. For explainable prediction on a per example basis, please run ML.EXPLAIN\_PREDICT or ML.EXPLAIN\_FORECAST.

Feature Name	Attribution
PropertyGFATotal	468,744.782
PropertyGFABuilding_s_	200,365.819
LargestPropertyUseTypeGFA	190,876.808
BuildingType	106,253.254
LargestPropertyUseType	101,570.498
NaturalGas_therms_	70,613.393
PrimaryPropertyType	46,614.349
Neighborhood	38,702.341
ListAllPropertyUseTypes	31,177.081
SecondLargestPropertyUseType	13,559.222
ThirdLargestPropertyUseType	6,502.598

<b>Mean absolute error</b>	376,870.2441
<b>Mean squared error</b>	1,188,105,880,684.7883
<b>Mean squared log error</b>	0.9301
<b>Median absolute error</b>	99,025.6015
<b>R squared</b>	0.7283

# Summary of the results for test data

---

Model / Y	R2	RMSE
Python: Random Forest (TotalGHGEmissions)	0.89	62.3
Python: XGB (Electricity(kWh))	0.88	950 769
BQ ML: DNN (TotalGHGEmissions)	0.94	159
BQ ML: Random Forest (Electricity(kWh))	0.73	1 090 002

# Vertex AI Auto ML

# Loading dataset

Vertex AI    [stage1\\_table\\_clean](#)    Notifications [NEW MODEL](#) [...](#)

TOOLS

- Dashboard
- Model Garden
- Workbench
- Pipelines

GENERATIVE AI STUDIO

- Overview
- Language
- Vision
- Speech

DATA

- Feature Store
- Marketplace

SOURCE ANALYZE  
Total rows: 3,35 /

General statistics generated by Jul 27, 2023 3:55 PM [GENERATE STATISTICS](#)

Filter Enter property name or value

Column name ↑	Missing % (count) ?	Distinct values ? ↑
BuildingType	-	The number of distinct values in this column
Electricity_kWh_	-	
LargestPropertyUseType	-	56
LargestPropertyUseTypeGFA	-	3113
ListOfAllPropertyUseTypes	-	463
NaturalGas_therms_	-	2109
Neighborhood	-	19
PrimaryPropertyType	-	24
PropertyGFABuilding_s_	-	3175
PropertyGFATotal	-	3179
SecondLargestPropertyUseType	-	52
ThirdLargestPropertyUseType	-	46
TotalGHGEmissions	-	2816

Related resources >

Training jobs and models

Use this dataset and annotation set to train a new machine learning model with AutoML or custom code. Selecting AutoML on Pipelines will create a Run on Vertex AI Pipelines. Run information will be found on the [Runs tab](#) under Pipelines.

[TRAIN NEW MODEL](#)

# Step by step pipeline: Regression model

The screenshot shows the Vertex AI interface for creating a pipeline run. The left sidebar has a 'Pipelines' section selected. The main area is titled 'Create pipeline run' and shows the 'Training method' step selected in a five-step process.

**Run details** (checked)

**Runtime configuration** (checked)

**Training method** (checked)

**Training options** (step 4)

**Compute and pricing** (step 5)

**SUBMIT**

**Dataset \***  
stage1\_table\_clean

**Objective \***  
Regression

**Target column \***  
TotalGHGEmissions

**Model name \***  
emissions\_model

**Description**

Create an evaluation on the trained model

Enable model distillation  
This reduces resulting model size

Skip architecture search  
This saves training time by skipping the architecture search step

**Data split**

Random  
Randomly assign your data for training, validation, and testing.

**Training \*** 80

**Validation \*** 10

**Testing \*** 10

# Precise train\_test\_split

**Vertex AI**

TOOLS

- Dashboard
- Model Garden
- Workbench
- Pipelines**

GENERATIVE AI STUDIO

- Overview
- Language
- Vision
- Speech

DATA

- Feature Store
- Marketplace

Create an evaluation on the trained model

Enable model distillation  
This reduces resulting model size

Skip architecture search  
This saves training time by skipping the architecture search step

**Data split**

Random  
Randomly assign your data for training, validation, and testing.

Training \*  Validation \*  Testing \*



- Training: 80%
- Validation: 10%
- Test: 10%

Manual  
You assign each data row for training, validation, and testing.

Chronological  
The earliest of your data is assigned to training, the next for validation and the latest for testing. This option requires a Time column in your dataset.

Stratified  
A variation of random split, which creates splits by preserving the percentage for each target class.

Export additional model without custom TensorFlow operators

# Transform data (change string to categorical)

The screenshot shows the Vertex AI Pipelines interface for transforming data. On the left, the sidebar includes sections for Vertex AI, TOOLS (Dashboard, Model Garden, Workbench), Pipelines (selected), GENERATIVE AI STUDIO (Overview, Language, Vision, Speech), and DATA (Feature Store, Marketplace). The main area has a sidebar titled "Runtime configuration" with checkboxes for "Training method" and "Training options" (which is selected). A "SUBMIT" button is visible. The main workspace is titled "specify the data types in your dataset. If unspecified, AUTOML will try to apply the most relevant transformation option." It features a "GENERATE STATISTICS" button and a "Filter" input field. Below is a table of columns with transformation settings:

Column name ↑	Transformation	⋮
BuildingType	Categorical	⊖
Electricity_kWh_		
LargestPropertyUseType		
LargestPropertyUseTypeGFA		
ListOfAllPropertyUseTypes		
NaturalGas_therms_		
Neighborhood		
PrimaryPropertyType		
PropertyGFABuilding_s_		
PropertyGFATotal		
SecondLargestPropertyUseType		
ThirdLargestPropertyUseType	Categorical	⊖
TotalGHGEmissions	Target	

A section titled "Customize transformation options" shows radio buttons for "Categorical" (selected), "Text", "Timestamp", and "Numeric". An "APPLY" button is at the bottom right. The right sidebar lists "Recommended for you" items like "Sample code", "Help document", "Pipelines sample notebook", "Introduction to Vertex AI", "Help document", and "Vertex Pipelines & ML".

Rows per page: 50 ▾

1 – 13 of 13 < >

# Define optimization objective

The screenshot shows the Vertex AI Pipelines interface. On the left, there's a sidebar with sections like TOOLS, GENERATIVE AI STUDIO, and DATA. The GENERATIVE AI STUDIO section is expanded, showing options like Overview, Language, Vision, and Speech. The DATA section is also expanded, showing Feature Store and Marketplace. The main area is titled "Define optimization objective". It includes a section for "Optimization objective \*", where "RMSE (Default)" is selected. Below it are sections for "Architecture search" (Number of parallel trials: 35), "Cross validation training" (Number of selected trials: 5), and "Additional settings" (Study spec override (JSON)).

**Vertex AI**

TOOLS

- Dashboard
- Model Garden
- Workbench
- Pipelines

GENERATIVE AI STUDIO

- Overview
- Language
- Vision
- Speech

DATA

- Feature Store
- Marketplace

**Optimization objective \***

RMSE (Default)  
Capture more extreme values accurately

MAE  
View extreme values as outliers with less impact on the model

RMSLE  
Penalize error on relative size rather than absolute value. Especially helpful when both predicted and actual values can be quite large. It is undefined when the predicted or ground truth is less than 0.

**Architecture search**

Number of parallel trials  ?

**Cross validation training**

Number of selected trials  ?

Number of parallel trainers  ?

**Additional settings**

Study spec override (JSON)

# Setting budget and compute type

The screenshot shows the Vertex AI interface for creating a pipeline run. The left sidebar includes links for Tools (Dashboard, Model Garden, Workbench, Pipelines), Generative AI Studio (Overview, Language, Vision, Speech), and Data (Feature Store, Marketplace). The main panel is titled "Create pipeline run" and lists five steps: Run details, Runtime configuration, Training method, Training options, and Compute and pricing. The "Compute and pricing" step is currently selected. It contains fields for "Budget \* 1 Maximum node hours" (with a question mark icon) and "Estimated completion: Jul 27, 2023 6 PM GMT+2". A toggle switch for "Enable early stopping" is turned on, with a descriptive text below it. Below these, sections for "Compute settings" (selecting worker pool type), "Architecture search tuner" (selecting trainer and evaluator machine types), and "Cross-validation trainer" (selecting trainer machine type) are visible.

Vertex AI

Create pipeline run

LEARN

TOOLS

- Dashboard
- Model Garden
- Workbench
- Pipelines

GENERATIVE AI STUDIO

- Overview
- Language
- Vision
- Speech

DATA

- Feature Store
- Marketplace

Run details

Runtime configuration

Training method

Training options

Compute and pricing

Budget \* 1 Maximum node hours ?

Estimated completion: Jul 27, 2023 6 PM GMT+2

Enable early stopping

Compute settings

Select the type of virtual machine to use for your worker pool

Architecture search tuner

Trainer machine type \* n1-standard-8, 8 vCPUs, 30 GiB memory

Evaluator machine type \* n1-standard-4, 4 vCPUs, 15 GiB memory

Cross-validation trainer

Trainer machine type \*

# Training process is longlasting

The screenshot shows the Vertex AI Training interface. On the left, there's a sidebar with sections for GENERATIVE AI STUDIO (Overview, Language, Vision, Speech), DATA (Feature Store, Datasets, Labeling tasks), and MODEL DEVELOPMENT (Training, Experiments). The 'Training' section is currently selected. At the top right, there are buttons for CREATE (+), REFRESH (refresh icon), and LEARN (graduation cap icon). Below the top navigation, there are tabs for TRAINING PIPELINES (selected), CUSTOM JOBS, HYPERPARAMETER TUNING JOBS, and NAS JOBS. A main content area contains a descriptive paragraph about training pipelines and a table of current jobs.

Training pipelines are the primary model training workflow in Vertex AI. You can use training pipelines to create an AutoML-trained model or a custom-trained model. For custom-trained models, training pipelines orchestrate custom training jobs and hyperparameter tuning with additional steps like adding a dataset or uploading the model to Vertex AI for prediction serving. [Learn More](#)

Name	ID	Status	Job type	Model type	Duration	Last update	⋮
<a href="#">electricity_test1</a>	8708724797472768000	<span>Training</span>	Training pipeline	Tabular regression	5 min 22 sec	Jul 28, 2023 2:58:20 P	⋮
<a href="#">test5</a>	7850507593481977856	<span>Finished</span>	Training pipeline	Tabular regression	1 hr 57 min	Jul 28, 2023 2:45:25 P	⋮

# Evaluation (TotalGHGEmissions)

Vertex AI

test5 < Version 1 > VIEW DATASET EXPORT LEARN

Vision EVALUATE DEPLOY & TEST BATCH PREDICT VERSION DETAILS

Speech untitled\_7897944772768981168 COMPARE CREATE EVALUATION

DATA

- Feature Store
- Datasets
- Labeling tasks

MODEL DEVELOPMENT

- Training
- Experiments
- Metadata

DEPLOY AND USE

- Model Registry
- Marketplace

EVALUATE

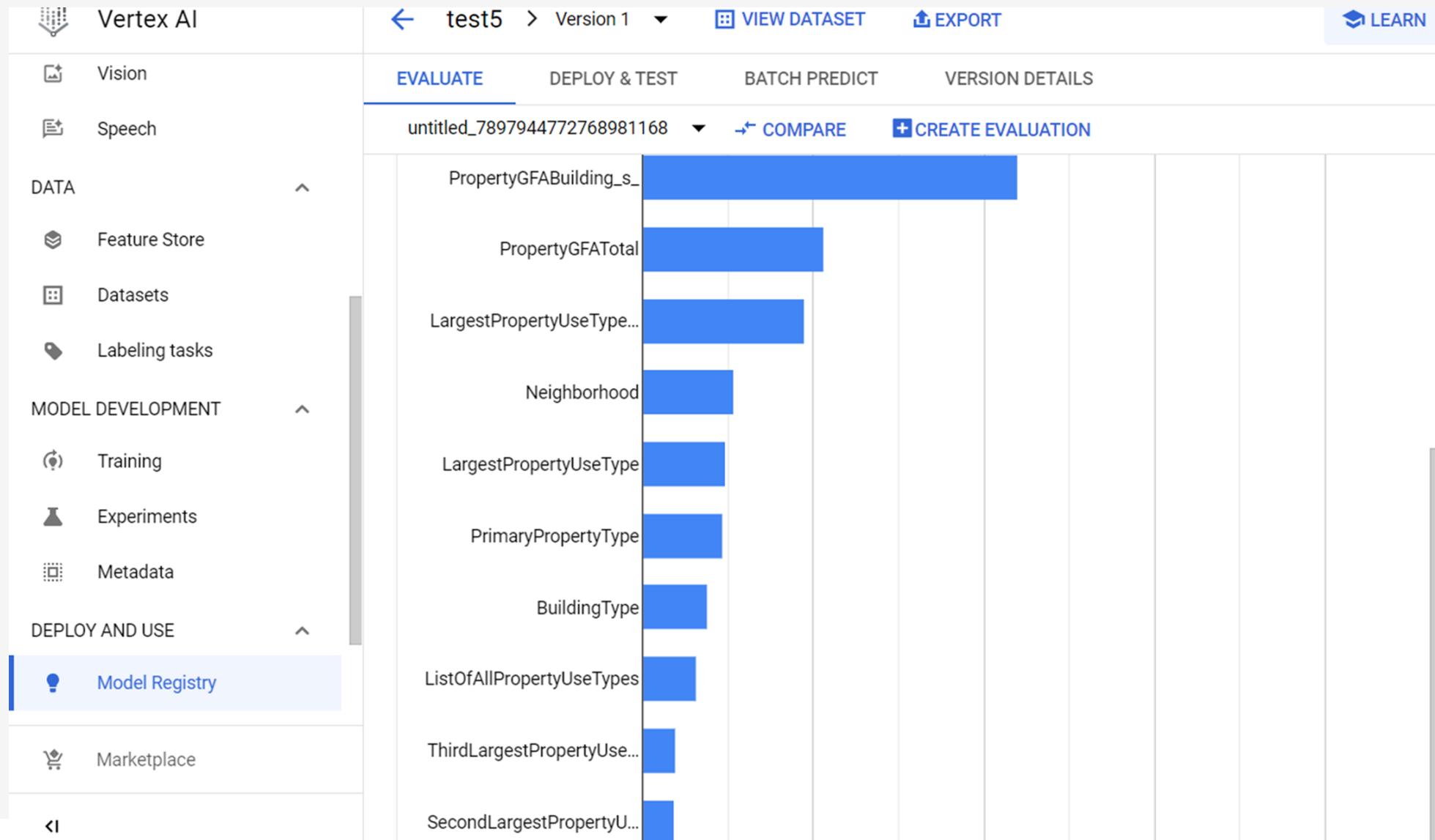
Target column	MAE	MAPE	RMSE	RMSLE	r^2
TotalGHGEmissions	39.201	39.574	142.116	0.47	0.939

Feature importance

Model feature attribution tells you how important each feature is when making a prediction. Attribution values are expressed as a percentage; the higher the percentage, the more strongly that feature impacts a prediction on average. Model feature attribution is expressed using the Sampled Shapley method. [Learn more](#)

Feature	Importance (%)
NaturalGas_therms	~45%
PropertyGFABuilding_s	~25%
PropertyGFA_Total	~15%
LargestPropertyUseType...	~10%

# Feature Importance (TotalGHGEmissions)



# Evaluation (Electricity(kWh))

Vertex AI

electricity\_test1 Version 1

VIEW DATASET EXPORT LEARN

EVALUATE DEPLOY & TEST BATCH PREDICT VERSION DETAILS

untitled\_8226014217102408013 ▾ ↗ COMPARE + CREATE EVALUATION

column	MAE ?	MAPE ?	RMSE ?	RMSLE ?	r^2 ?
city_kWh_	396,109.56	58.227	<b>964,808.4</b>	0.587	0.839

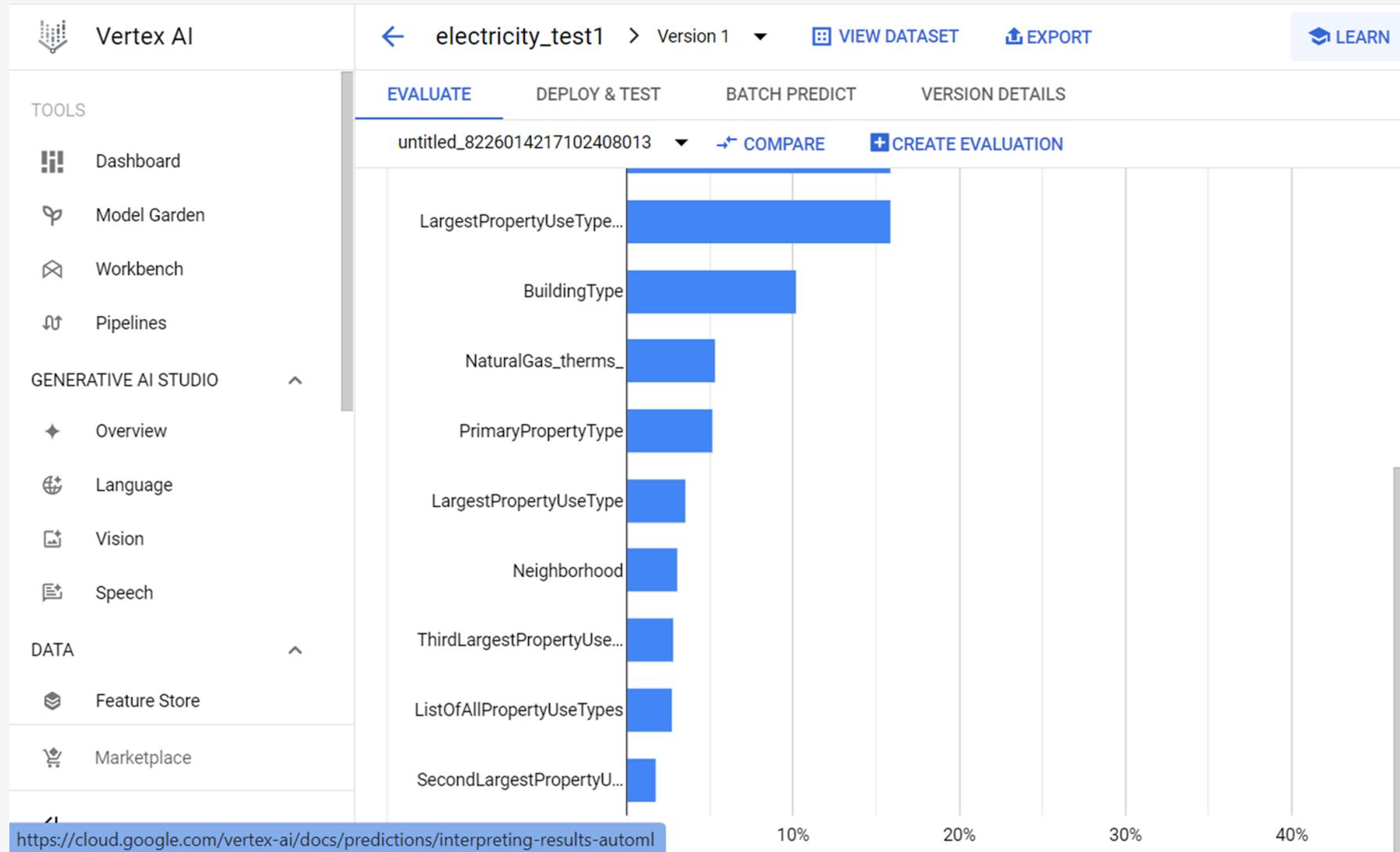
Feature importance

Model feature attribution tells you how important each feature is when making a prediction. Attribution values are expressed as a percentage; the higher the percentage, the more strongly that feature impacts a prediction on average. Model feature attribution is expressed using the Sampled Shapley method. [Learn more](#)

The chart displays the importance of various features in predicting electricity consumption (kWh). The features and their approximate importance percentages are:

Feature	Importance (%)
PropertyGFATotal	~45%
PropertyGFABuilding_s...	~15%
LargestPropertyUseType...	~12%
BuildingType	~8%
NaturalGas_therms	~5%

# Feature Importance (Electricity(kWh))



# Summary of the results for test data

---

Model / Y	R2	RMSE
Python: Random Forest (TotalGHGEmissions)	0.89	62.3
Python: XGB (Electricity(kWh))	0.88	950 769
BQ ML: DNN (TotalGHGEmissions)	0.94	159
BQ ML: Random Forest (Electricity(kWh))	0.73	1 090 002
Vertex AI Auto ML: Tabular Regression (TotalGHGEmissions)	0.94	142
Vertex AI Auto ML: Tabular Regression (Electricity(kWh))	0.84	964 808

# Model Deployment (example)

# Deploy the model (placed in Bucket)

← emission\_model [EDIT SETTINGS](#) [SAMPLE REQUEST](#)

Region	Logs	Model Monitoring
us-central1	<a href="#">View Logs</a>	Disabled

<input checked="" type="checkbox"/> Model	Status	Deployment resource pool	Most recent alerts	Monitoring	Traffic split	Compute nodes	Type	Created
<input checked="" type="checkbox"/> <a href="#">test5</a> <a href="#">(Version 1)</a>	<span>✓ Ready</span>	–	–	Disabled	100%	Auto (1 minimum, 1 maximum)	Tabular	Jul 31, 2023, 12:12:41 PM

[DEPLOY ANOTHER MODEL](#)

Chart interval: ✓ 1 hour 6 hours 12 hours 1 day 2 days 4 days 7 days 14 days 30 days

[PERFORMANCE](#)

[RESOURCE USAGE](#)

# Automatic model test

← test5 → Version 1 ▾ [VIEW DATASET](#) [EXPORT](#)

EVALUATE **DEPLOY & TEST** BATCH PREDICT VERSION DETAILS

**Test your model** [PREVIEW](#)

Feature column name	Type	Value	Local feature im
BuildingType	Text	NonResidential	0
PrimaryPropertyType	Text	Low-Rise Multifamily	0
Neighborhood	Text	DOWNTOWN	0
PropertyGFATotal	Text	44416.0	0
PropertyGFABuilding_s_	Text	43330.0	0
ListofAllPropertyUseTypes	Text	Multifamily Housing	0
LargestPropertyUseType	Text	Multifamily Housing	0

**Predict label**  
28.776296615600586

**Prediction result**  
[12.25694942474365, 70.51982116699219]

**95% prediction interval**

# Workbench request to the model

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** test\_endpoint.ipynb
- Toolbar:** Includes icons for file operations (New, Open, Save, etc.), Code, Execute, and Python (Local) kernel selection.
- Code Cells:**
  - [1]: `from google.cloud import aiplatform`
  - [2]: `project="733376066474"`  
`endpoint_id="4941851768062476288"`  
`location="us-central1"`  
`endpoint_name=f"projects/{project}/locations/{location}/endpoints/{endpoint_id}"`  
`endpoint=aiplatform.Endpoint(endpoint_name=endpoint_name)`
  - [3]: `instances=[{'BuildingType': 'NonResidential', 'PrimaryPropertyType': 'Low-Rise Multifamily', 'Neighborhood': 'DOWNTOWN', 'PropertyGFATotal': '44416.0', 'PropertyGFABuilding_s_': '43330.0', 'ListOfAllPropertyUseTypes': 'Multi Family Residential', 'LargestPropertyUseType': 'Multifamily Housing', 'LargestPropertyUseTypeGFA': '39900.0', 'SecondLargestPropertyUseType': 'Parking', 'ThirdLargestPropertyUseType': 'Retail Store', 'NaturalGas_therms_': '3261.589844'}]`
  - [4]: `endpoint.predict(instances=instances)`
- Status Bar:** Shows the status of the last cell execution: [4]: Prediction(...)