Solar irradiance classification

Dmitrii Kuzimin, Yaroslava Bryukhanova, Timofey Brayko, Artemii Miasoedov, Artur Rakhmetov

1. Introduction

Weather prediction is an extremely important industry in modern life. Weather forecasts are used not only to decide what clothes to wear each day—they also impact agriculture, construction, transportation, and the energy sector.

To make accurate predictions, numerous weather stations collect information about the Earth’s current conditions. These stations consist of various sensors. Some are simple, like thermometers, while others are more complex and involve moving parts, such as anemometers (tools used to measure wind speed). As a result, advanced weather stations require significant space for installation and frequent maintenance, including cleaning sensors, lubrication, and more.

We propose a novel idea: creating a compact weather station using inexpensive sensors. Much like CubeSats—small, low-cost alternatives to expensive satellites – these stations could be mass-produced, easily installed, and quickly replaced. We plan to use the collected data to infer other parameters that are not directly measured.

To test this concept, we decided to infer solar irradiance (solar radiation) using other parameters such as temperature, time, and ozone levels. Solar irradiance is the amount of solar energy reaching the Earth's surface and is a key metric in agriculture, construction, and energy production. It affects:

- crop growth and vegetation
- building insulation performance
- solar power plant efficiency

Solar irradiance is typically measured using solar panels or pyranometers. However, these devices can become dirty over time, leading to inaccurate readings. Our approach uses data from simpler, more robust devices—such as thermometers and clocks—to estimate solar irradiance. If successful, this pipeline could be extended to infer other environmental properties, potentially simplifying weather station installation and maintenance while also improving weather forecasts.

2. Data description

The **Ozone Training Data** dataset, curated by Geoweaver [link], comprises atmospheric and environmental measurements intended for training machine learning models to predict ozone levels. It encompasses various meteorological and chemical variables collected across different locations and time periods. The dataset is suitable for applications in air quality monitoring, environmental studies, and predictive modeling.

Dataset Overview:

- **Size:** Approximately 9.29 million rows (1.5 GB)
- **Format:** CSV
- **License:** CC-BY-4.0
- **Language:** English
- **Year:** 2024
- **Authors:** Ziheng Sun, Yunyao Li, Daniel Tong, Siqi Ma

Key Features:

The dataset includes the following columns:

Column Name	Unit	Data Type	Description
Latitude	Degrees	Float64	Latitude of the observation location.
Longitude	Degrees	Float64	Longitude of the observation location.
CMAQ12KM_O3(ppb)	ppb	Float64	Ozone concentration in parts per billion (CMAQ model data).
CMAQ12KM_NO2(ppb)	ppb	Float64	Nitrogen dioxide concentration in parts per billion (CMAQ model data).
CMAQ12KM_CO(ppm)	ppm	Float64	Carbon monoxide concentration in parts per million (CMAQ model data).
CMAQ_OC(ug/m3)	μg/m^3	Float64	Organic carbon concentration in micrograms per cubic meter (CMAQ model data).
CO(moles/s)	Moles/second	Float64	Carbon monoxide emission rate.
PRSFC(Pa)	Pascals	Float64	Surface pressure.

Column Name	Unit	Data Type	Description
PBL(m)	Meters	Float64	Planetary boundary layer height.
TEMP2(K)	Kelvin	Float64	Temperature at 2 meters above ground level.
WSPD10(m/s)	m/s	Float64	Wind speed at 10 meters above ground level.
WDIR10(degree)	Degrees	Float64	Wind direction at 10 meters above ground level.
RGRND(W/m2)	W/m²	Float64	Ground solar radiation.
CFRAC	-	Float64	Cloud fraction.
month	-	Integer	Month of the observation (1-12).
day	-	Integer	Day of the observation (1-31).
hours	-	Integer	Hour of the observation (0-23).

Target Label:

The target variable for classification is **RGRND(W/m2)**, representing ground solar radiation. Although originally a continuous variable, it can be discretized into categorical classes (e.g., low, medium, high) to suit classification tasks. The chosen classes are:

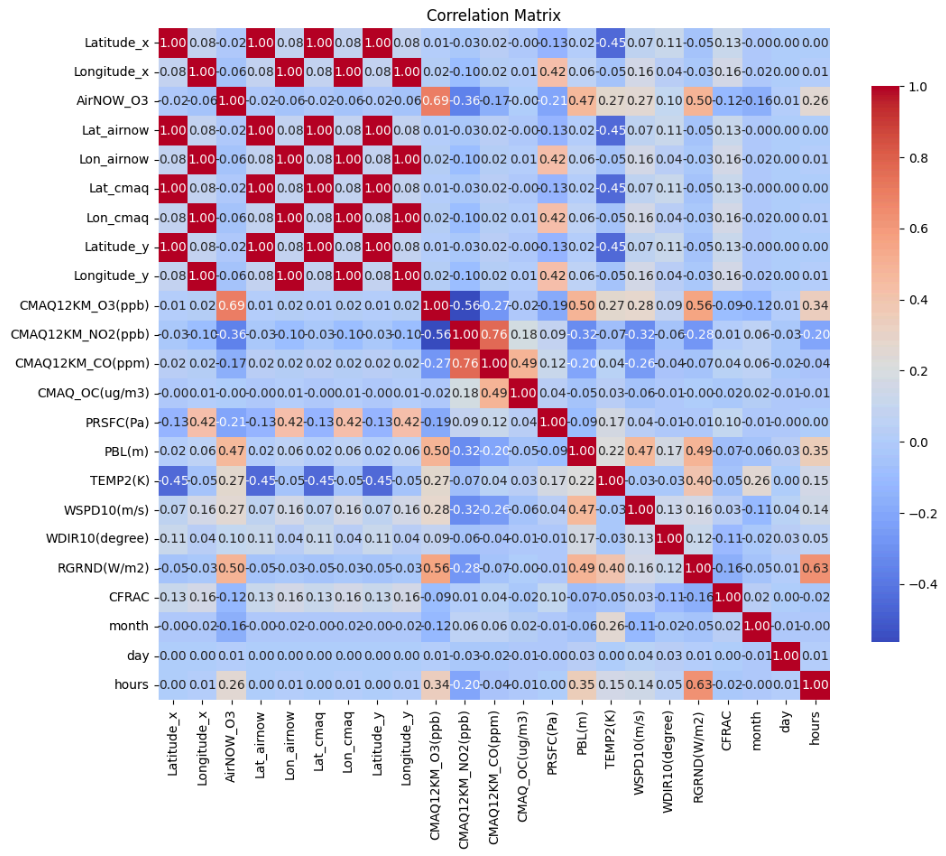
- Low: 0 - 100 W/m^2
- Medium: 100 - 500 W/m^2
- High: > 500 W/m^2

3. Architecture of data pipeline

The data processing and analysis pipeline is structured through several key stages, from initial data acquisition to feature engineering for model training.

3.1. Data Loading and Preprocessing

The dataset, `training_data.csv`, was loaded directly from HuggingFace.
At the initial preprocessing step we constructed a correlation matrix to identify and eliminate highly correlated or redundant features. We've revealed that several columns related to geographical coordinates were duplicates: `Latitude_x, Lat_airnow, Lat_cmaq, Latitude_y, and Longitute_x, Lon_airnow, Lon_cmaq, Longitude_y.`
Redundant coordinate columns were removed, keeping Latitude_x and Longitute_x as the primary geographical coordinates.



After the data was splitted and saved within CSV files.

- **stations.csv**: Contains station_id and its corresponding coordinates (Latitude_x, Longitude_x).
- **records.csv**: Contains all other observational data columns as detailed in the ***Data Description*** Section.

3.2. Data Ingestion into SQL Database

At this stage SQL tables: *records* and *stations*, were created. Python scripts were implemented to create and transfer data from prepared CSV files.

Both previous steps are implemented in ***scripts/build_database.py.***

3.3. Data Transfer to Hadoop (HDFS) using Parquet

For scalable big data processing, the data residing in the SQL database was transferred to HDFS. Apache Parquet was selected as the storage format.
The data transfer was accomplished using Sqoop with the following command:

```
sqoop import-all-tables \
   --connect jdbc:postgresql://hadoop-04.uni.innopolis.ru/team29_projectdb \
   --username team29 -P \
   --compression-codec=snappy \
   --compress \
   --as-parquetfile \
   --warehouse-dir=project/warehouse \
   --m 1
```

This command imports all tables from our PostgreSQL database, compresses them using Snappy, and stores them as Parquet files in the HDFS warehouse directory.

3.4. Hive Integration

Apache Hive was utilized to impose a schema on the Parquet files stored in HDFS, enabling SQL-like querying and facilitating data analysis at scale. External Hive tables were created, pointing to the Parquet data in the `project/warehouse` HDFS directory.

Creation of External Tables:
To avoid data movement and retain a single source of truth within HDFS, external Hive tables were created. These tables directly reference the Parquet files previously imported via Sqoop and stored under the project/warehouse directory in HDFS.

3.5. Feature Engineering in PySpark

The final stage before model training involved feature engineering using PySpark, leveraging its distributed processing capabilities.
The following transformations were applied:

- **Numerical Feature Scaling:** Features such as Ozone, NO2, CO, OC, Pressure, PBL, Temperature, Wind Speed, and Cloud Fraction were standardized using a StandardScaler.

$$z = \frac{x - \mu}{\sigma}$$

- **Cyclical Feature Transformation:** Timel features like month, day, and hour were transformed using a custom SinCosTransformer. This technique encodes cyclical features into two dimensions using sine and cosine functions, preserving their cyclical nature according to the following formulas:

$$sin(\frac{2 \times \pi \times x}{X_{max}}) \qquad cos(\frac{2 \times \pi \times x}{X_{max}})$$

- **Coordinate Transformation:** longitude and latitude were transformed into the Earth-Centered, Earth-Fixed (ECEF) coordinate system. This transformation provides a Cartesian coordinate representation. A custom class also was implemented to handle this.

$$X = (N + h)\cos\phi\cos\lambda$$
$$Y = (N + h)\cos\phi\sin\lambda$$
$$Z = \left(\frac{b^2}{a^2}N + h\right)\sin\phi$$

- **Label Discretization:** Target variable of ground solar radiation was categorized into 'low', 'medium', and 'high' based on the predefined thresholds:

$$Label = \begin{cases} 0, & \text{Radiation} \leq 100 \\ 1, & 100 < \text{Radiation} \leq 500 \\ 2, & \text{Radiation} > 500 \end{cases}$$

  Custom class for transformations also was implemented.

4. Data Preparation

4.1. Entity-Relationship Diagram (ER Diagram)

The database schema after splitting the data consists of two primary tables: stations and records.

- stations **Table:**
  - `station_id` (Primary Key): Unique identifier for each observation station.
  - `Latitude_x`: Latitude coordinate of the station.
  - `Longitude_x`: Longitude coordinate of the station.
- records **Table:**
  - `record_id`: Unique identifier for each observation record.
  - `station_id`: Links the record to a specific station.
  - Other columns were renames for simplicity: `airnow_ozone` `cmaq_ozone`, `cmaq_no2`, `cmaq_co`, `cmaq_oc`, `pressure`, `pbl`, `temperature`,

`wind_speed`, `wind_direction`, `radiation`, `cloud_fraction`, `month`, `day`, `hour`. Observational data as described in Data Description.

ER diagram shows the relationship between *stations* and *records* tables as one-to-many: one station can have many records.

4.2. Example of Sations table:

Identifier	**Latitude**	**Longitude**
120350004	29.489082	-81.276833
360850111	40.5802	-74.199402
390610040	39.12886	-84.504044
840450070006	34.63596	-82.810669
60712002	34.100132	-117.491982

4.3. Example of Records table:

record_id	1	2	3
station_id	120350004	360850111	390610040
airnow ozone	35	52	41
cmaq ozone	25	45	45
cmaq no2	0	2	1
cmaq co	84	147	137
cmaq oc	1	2	5
pressure	101384	100797	98891
pbl	1319	1163	2170
temperature	305	299	302
wind speed	6	4	6
wind direction	242	189	339
radiation	576	416	696
cloud fraction	0	0	0
month	8	8	8
day	1	1	1
hour	21	21	21

4.4. HIVE Tables

To efficiently support large-scale, SQL-like data analysis on HDFS data, Hive was used to define and manage structured tables on top of Parquet files. The table schema and configuration were designed to support scalable querying and efficient data access.

Stations Table:

```
CREATE EXTERNAL TABLE stations (
   id STRING,
   latitude DOUBLE,
   longitude DOUBLE
)
STORED AS PARQUET
LOCATION 'project/warehouse/stations';
```

Purpose: Stores the static metadata of each station including its ID and geolocation.
No Partitioning: Since the station metadata is small and non-temporal (does not grow over time), partitioning is unnecessary.
Storage Format: Parquet was used for its efficient columnar storage and built-in compression capabilities.

Records Table Optimization:

1. Partitioning Strategy

```
PARTITIONED BY (month INT, day INT)
```

Why exactly month and day?

- **Temporal Partitioning**: Partitioning the records table by month and day allows for efficient querying of time-based data

- **Query Performance:** This strategy enables partition pruning, where Hive can skip irrelevant partitions during query execution, reducing I/O and improving performance

- **Data Management:** Organizing data by date simplifies data maintenance tasks, such as archiving or deleting old data.

- **Avoiding Over-Partitioning:** Choosing month and day strikes a balance between granularity and manageability, avoiding the overhead associated with too many small partitions.

2. **Bucketing Strategy**

```
CLUSTERED BY (station_id) INTO 2 BUCKETS
```

- **Efficient Joins**: Bucketing by station_id ensures that records with the same station ID are placed in the same bucket, optimizing join operations on this key.

- **Data Sampling**: Bucketing facilitates efficient sampling of data, which can be useful for exploratory data analysis and testing.

- **Parallel Processing**: Dividing data into buckets allows for parallel processing of data during query execution, improving performance.

- **Number of Buckets:** Choosing 2 buckets was based on the expected distribution of station_id values and the available computational resources, aiming for a balance between parallelism and overhead.

Hive Configuration Settings

The following Hive settings were configured to support dynamic partitioning, enforce bucketing, and optimize Tez execution:

```
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;
SET hive.enforce.bucketing=true;
SET hive.exec.max.dynamic.partitions=5000;
SET hive.exec.max.dynamic.partitions.pernode=5000;
SET hive.tez.container.size=4096;
SET parquet.memory.min.allocation.size=2097152;
SET hive.tez.auto.reducer.parallelism=true;
SET parquet.block.size=134217728;
```

- hive.exec.dynamic.partition=true & mode=nonstrict: Enable automatic creation of all partitions during data insertion.

- hive.enforce.bucketing=true: Ensures bucketing is applied, improving join and sampling performance.

- hive.exec.max.dynamic.partitions & .pernode = 5000: Prevents overload by limiting the number of partitions created globally and per node.

- hive.tez.container.size=4096: Allocates 4 GB memory per Tez container for efficient processing.

- parquet.memory.min.allocation.size=2097152: Ensures efficient Parquet write operations with a 2 MB minimum memory block.

- hive.tez.auto.reducer.parallelism=true: Allows Tez to auto-tune reducer parallelism for optimal query speed.

- parquet.block.size=134217728: Sets Parquet block size to 128 MB for better HDFS and I/O efficiency

5. Data Analysis

The analysis of station-based environmental data reveals several key patterns and insights regarding data distribution, geographic characteristics, and climatic relationships:

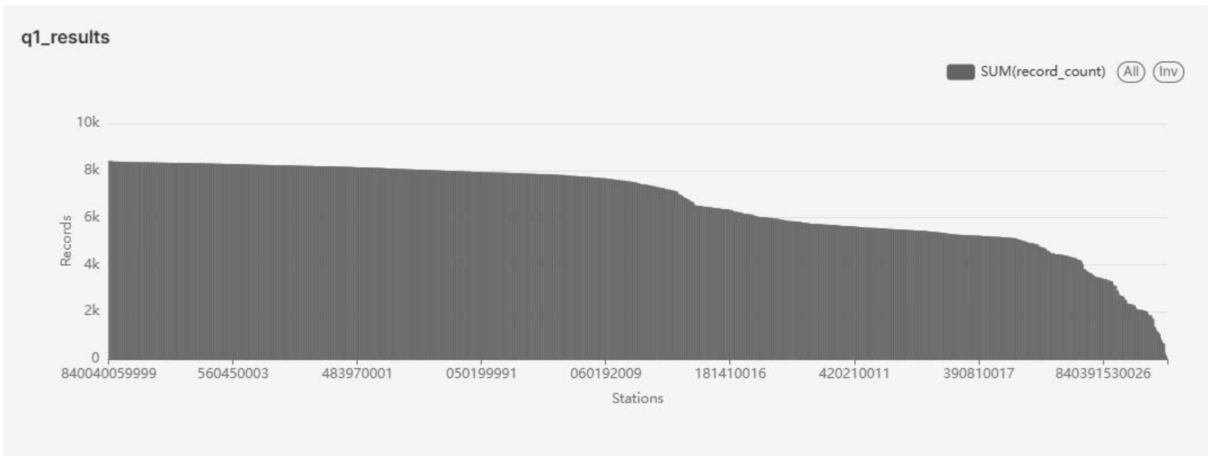
1) Query 1

The HQL query calculates the total number of records for each station_id and orders them in descending order.

The bar chart visualizes the distribution of record counts per station. It clearly shows that a few stations have a very high number of record

s (e.g., the leftmost bars are significantly taller), while the majority of stations have progressively fewer records.

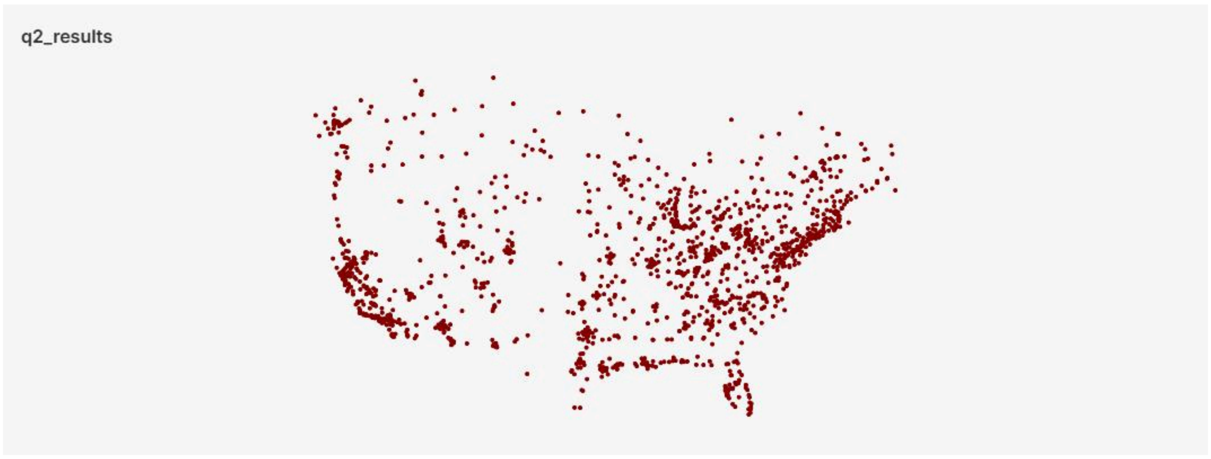
This indicates an uneven distribution of data collection or reporting frequency across the different stations. The station with station_id starting 840040059999 has the most records, close to 17,000.



2) Query 2

The HQL query selects the id, latitude, and longitude for all stations.

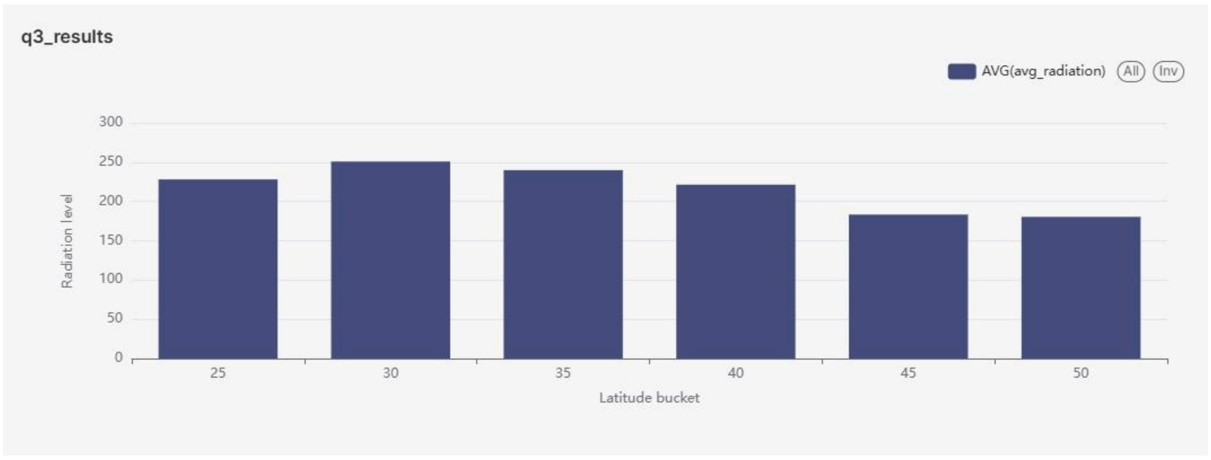
The scatter plot displays these coordinates, effectively mapping the station locations. The distribution of points shows that the stations are primarily located within the contiguous United States, with denser clusters in the eastern half, particularly the Northeast and Southeast, and along the West Coast.



3) Query 3

The HQL query calculates the average solar radiation by grouping stations into 5-degree latitude buckets. The bar chart displays this average radiation (W/m^2) for each latitude bucket.

This pattern suggests that, on average, locations in the lower mid-latitudes (around 25-35 degrees) represented in this dataset receive more solar radiation than those at higher latitudes (40-50 degrees). This is consistent with general climatological expectations where lower latitudes tend to receive more direct sunlight.

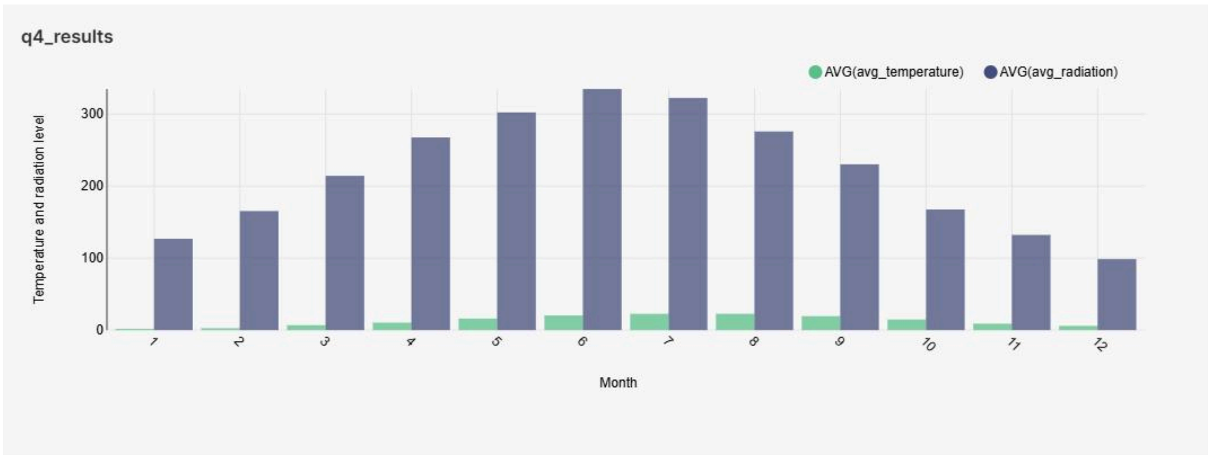


4) Query 4

The HQL query calculates the average temperature and average radiation for each month of the year. The x-axis is ordered from 1 down to 12.

The bar chart shows that average temperature remains relatively high and consistent across the displayed months. Average radiation shows more variation, with peaks around mid-year (e.g., month 6, 7) and lower values at the start/end of this period (month 1, 12).

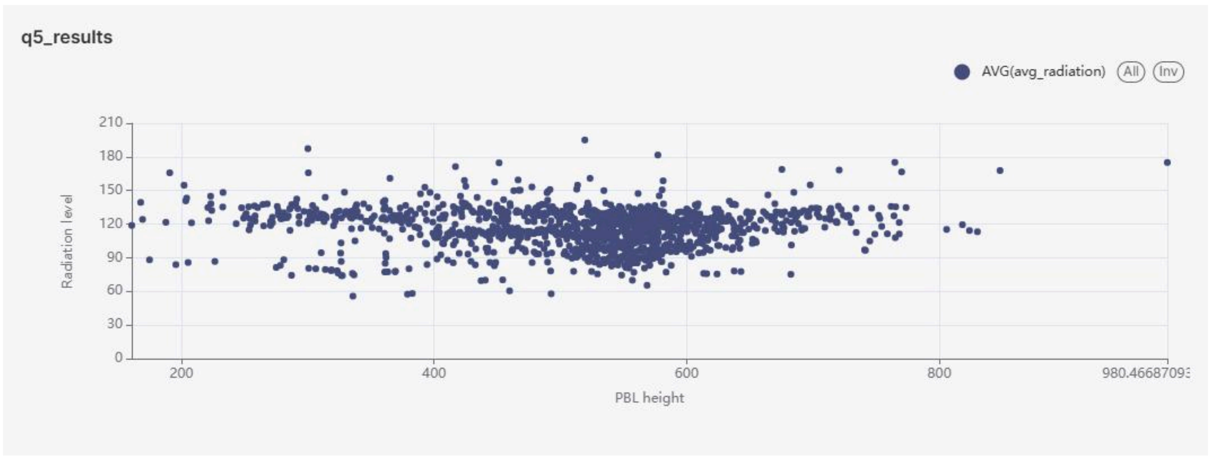
This suggests a typical diurnal cycle for radiation, though the hourly aggregation might smooth out finer details.



5) Query 5

The HQL query calculates the average Planetary Boundary Layer height and average radiation for each station_id.

The scatter plot displays avg_radiation on the y-axis against avg_pbl on the x-axis for each station. The plot suggests that there isn't a strong, simple linear relationship between average PBL height and average radiation across all stations. Most stations are clustered with avg_pbl values roughly between 100 and 150, and avg_radiation values mostly between 150 and 350. There is one notable outlier station with a very high avg_pbl (around 980) and a relatively high avg_radiation (around 350). The density of points is highest in the center of this cluster.

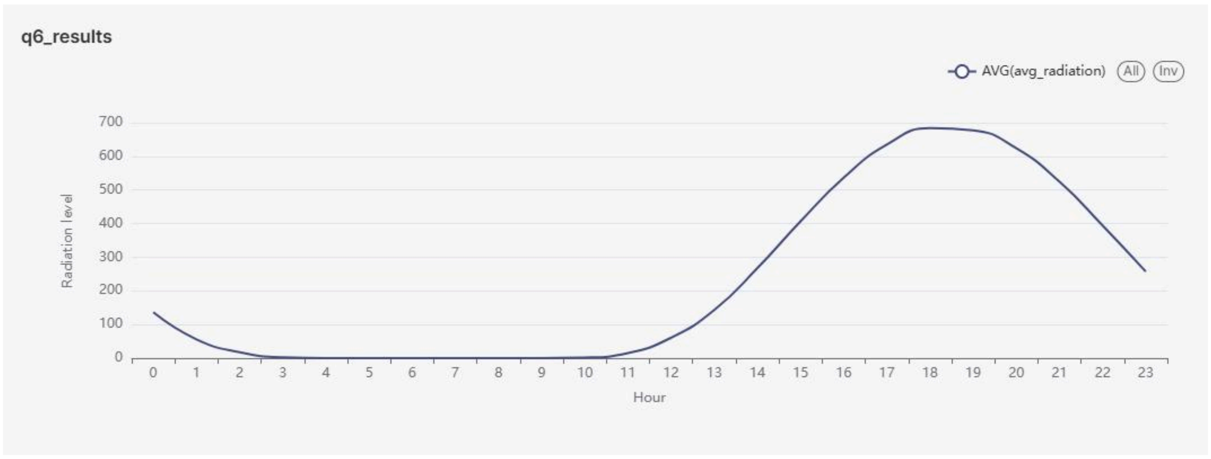


6) Query 6

The HQL query calculates the average pbl, cmaq_ozone, and radiation for each hour of a day (presumably).

The line chart shows the trend of average solar radiation over the hours of a day. It starts at a moderate level, dips in the first few hours (around hour 3-10), then rises to a peak around hour 17-20, followed by a sharp decline to the moderate level towards the end of the 24-hour period.

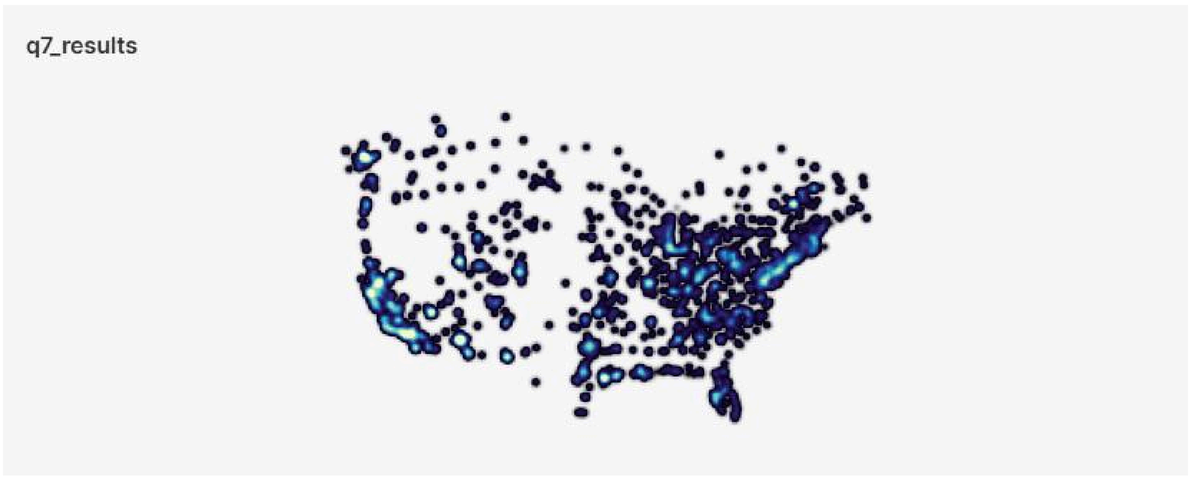
This suggests a period of Sun rises and higher solar intensity mid-day.



7) Query 7

The HQL query calculates the Pearson correlation coefficient between daily average temperature and solar radiation for each individual station, identified by its latitude and longitude. The resulting heatmap visualizes the strength of this correlation across the geographical distribution of these stations, within the United States.

Overall, the visualization highlights that while a positive correlation between solar radiation and temperature is generally expected, the strength of this relationship varies considerably by geographic location, reflecting diverse climatic conditions and environmental factors across the monitored stations.

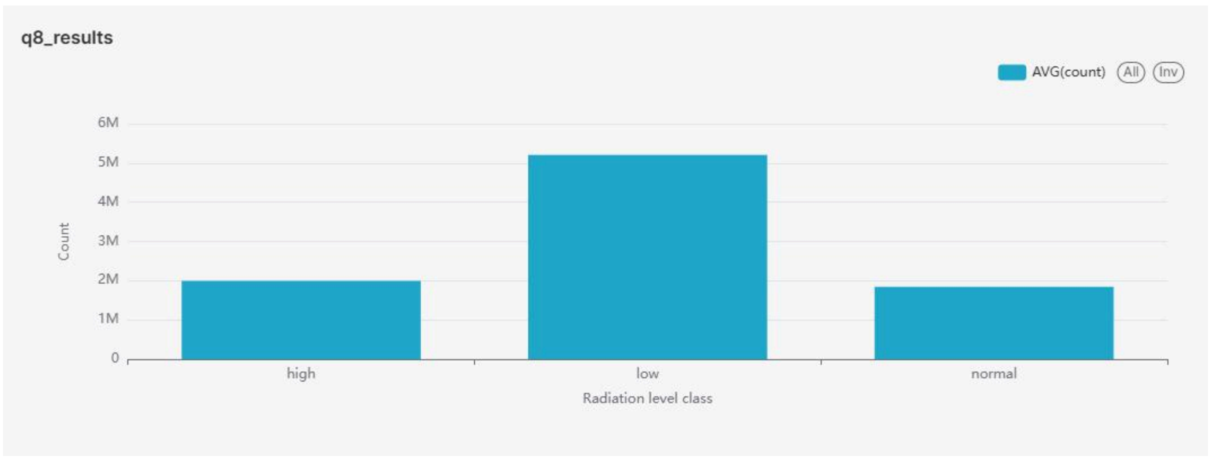


8) Query 8

The HQL query categorizes each record in the records table based on its radiation value into one of three levels:

'low': radiation <= 100 'normal': radiation > 100 AND radiation <= 500 'high': radiation > 500

This distribution suggests that the dataset is predominantly characterized by periods of low solar radiation. While "normal" radiation levels are the least frequent, there's still a substantial number of instances with "high" radiation. This could imply that conditions often tend towards lower radiation values, with less frequent occurrences of moderate levels, but a notable number of high radiation events.



Conclusion:

The data analysis reveals an uneven distribution of records across stations, with a few contributing the majority of data. Most stations are located in the contiguous U.S., especially in the East and along the coasts. Solar radiation is highest in mid-latitude regions (25–35°), consistent with expected climate patterns. Seasonal and daily trends show radiation peaking in summer months and late afternoons. The correlation between temperature and solar radiation varies by location, reflecting regional climatic differences.

6. ML Modeling

Data Preprocessing

Out of 17 existing features 15 have been selected for model training with 2 being dropped:

- airnow_ozone - ozone levels as cmaq_ozone but from a different model (AirNow).
- wind_direction - shown to have little effect on solar irradiance.

10 features are linear, starting from 0, having undefined upper bound, except cloud_fraction:

- radiation
- cmaq_ozone
- cmaq_no2
- cmaq_co
- cmaq_oc
- pressure
- pbl
- temperature
- wind_speed
- cloud_fraction

Thus, standard scaling has been applied to each of these features individually.

Other 5 features are periodic; 3 - representing time, 2 - geographic location:

- month
- day
- hour

- latitude
- longitude

Sin-cos representation was used for time data and ECEF representation – for geographic.

Original target radiation values have been binned into 3 categories:

- low < 100
- mild < 500
- high

Modelling

5 different classification models were considered for the project:

- Logistic regression
- Multilayer perceptron
- Naive bayes
- Random forest
- One-vs-Rest Linear SVC

Each model was trained using grid search and 3-fold cross validation.

Logistic regression

Grid search:

- `regParam = [0.01, 0.1, 1.0]`
- `elasticNetParam = [0.0, 0.5, 1.0]`
- `threshold = [0.3, 0.5, 0.7]`

Grid size: 27
K-fold: 3
Model fits: 27 * 3 = 81

Multilayer perceptron

Grid search:

-
  ```
  layers = [
      [features, 8, labels],
      [features, 16, labels],
      [features, 24, labels],
  ]
  ```
- `stepSize = [0.01, 0.05, 0.1]`
- `solver = ["l-bfgs", "gd"]`

Grid size: 18
K-fold: 3
Model fits: 18 * 3 = 54

Naive Bayes

Grid search:

- `smoothing = [0.5, 1.0, 1.5]`

-
  ```
  thresholds, [
      [1.0, 1.0, 1.0], # Neutral (no weighting)
  ```

```
        [1.5, 1.0, 0.5], # Favor class 0
        [0.5, 1.0, 1.5], # Favor class 2
    ]
```
Grid size: 9
K-fold: 3
Model fits: 9 * 3 = 27

Random Forest

Grid search:

- `numTrees = [16, 32, 64]`
- `maxDepth = [3, 5, 8]`
- `featureSubsetStrategy = ["auto", "sqrt", "log2"]`

Grid size: 27
K-fold: 3
Model fits: 27 * 3 = 81

Linear SVC

Grid Search

- `regParam = [0.01, 0.1, 1.0]`
- `aggregationDepth = [1, 2]`

Grid size: 6
K-fold: 3
Model fits: 6 * 3 = 18

Training

The dataset consists of ~9.92M rows. It was split into train/test subsets with 70% for train and 30% for test. f1 was used as a metric for best model selection. Overall training took 6 hours to complete.

Evaluation

Accuracy and F1 score were used to evaluate the models. The metrics were calculated on the test set for the best-performing models:

Model	Accuracy	F1
Logistic Regression	0.71	0.67
Naive Bayes	0.62	0.56
Random Forest	**0.85**	**0.84**
Multilayer Perceptron	0.57	0.42
Linear SVC	0.70	0.62

Among all the models evaluated, the random forest demonstrated the highest accuracy and F1 scores. The optimal hyperparameters identified for this model were:

- `numTrees = 64`
- `maxDepth = 8`
- `featureSubsetStrategy = "auto"`

Notably, both numTrees and maxDepth were set to the maximum values tested in the grid search. This suggests that further increasing these parameters could potentially enhance the model's performance in future experiments.

Deep Learning Requirements

During our final presentation, some concerns came up about whether we met all the project requirements. The main confusion was about our fifth team member – the Deep Learning Engineer. The issue was that we used Deep Learning with PySpark, while PyTorch was expected.

After the meeting, we did try to get PyTorch working with PySpark. We tested two tools:

- Horovod
- SparkTorch

Both are no longer properly supported from 2023. We managed to install Horovod (after a lot of trial and error). To even get that running, we had to disable parts of it and force TensorFlow instead of PyTorch, because:

- The required compiler version (g++5) isn't on the server (we only have g++4.8.5).
- PyTorch won't run unless we upgrade to g++4.9 – which we can't do.
- Gloo backend is broken in the old version, so we had to turn it off.

Then it failed again – missing libhdfs.so. We checked and found Horovod needs PySpark version 2.3.2, but going down from 3.2.4 caused more Spark errors due to version mismatches. So Horovod was a dead end. Same deal with SparkTorch – just didn't work.

So yes, we didn't use PyTorch. We still put in serious work, made things run, and tried to meet the project goals. We really hope this doesn't cost us a big penalty over a misunderstanding.

7. Data Presentation

In the dashboard we present structured visualization of the dataset characteristics, key insights and model results which we got from predictive modeling of solar irradiance levels. Dashboard divided into three parts:

1. Data Description - various metadata along with data preview
2. Data Insights - data characteristics of different kind
3. Model Scores - showcase performance of each model

For better understanding of this part, we provide a description of each chart.

-Data Description

- **Number of Samples:** this chart presents a solid single number, amount of samples in millions.
- **data_example:** presents 10 samples of the initial dataset with features like latitude, longitude, pressure, and radiation.

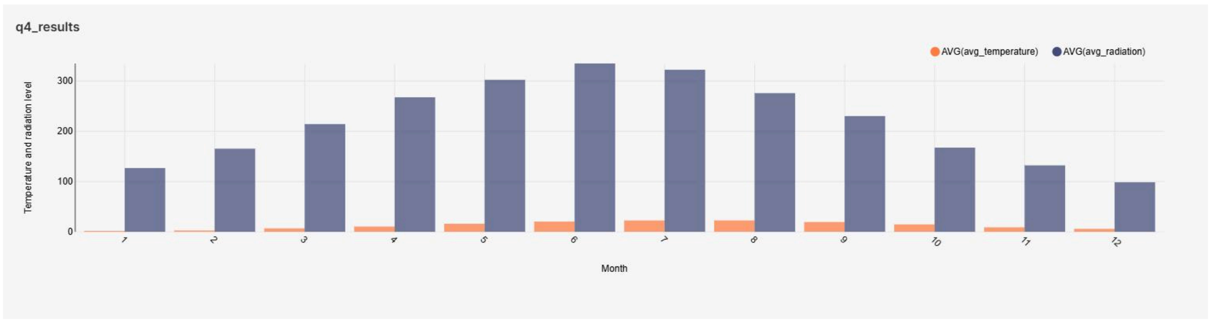
-Data Insights

- **q1_results:** Shows total number of records for each station_id in descending order.
- **q2_results:** The scatter plot maps the geographic locations of all stations using latitude and longitude. It shows that stations are primarily clustered within the United States, with higher density in the East and along coastal regions.
- **q3_results:** The bar chart groups stations by 5-degree latitude bins and shows the average solar radiation for each group. It demonstrates that mid-latitude zones (~25–35°) receive higher radiation on average.
- **q4_results:** The dual-axis bar chart shows average monthly temperature and solar radiation. While temperature remains steady across months, radiation peaks around summer months, following seasonal solar patterns.
- **q5_results:** The scatter plot compares average Planetary Boundary Layer (PBL) height and average radiation per station. Most stations are tightly clustered, showing no clear linear trend, though one outlier shows unusually high values.
- **q6_results:** The line chart shows average solar radiation across 24 hours. Radiation is low overnight, then rises and peaks in the late afternoon, reflecting a typical daily solar cycle.
- **q7_results:** The heatmap visualizes the correlation between average daily temperature and solar radiation across stations. It highlights regional differences, with correlation strength varying geographically due to local climate patterns.
- **q8_results:** The bar chart shows the distribution of radiation levels categorized as low, normal, or high. The majority of records fall under low radiation, while high levels occur more often than normal, suggesting skewed conditions.

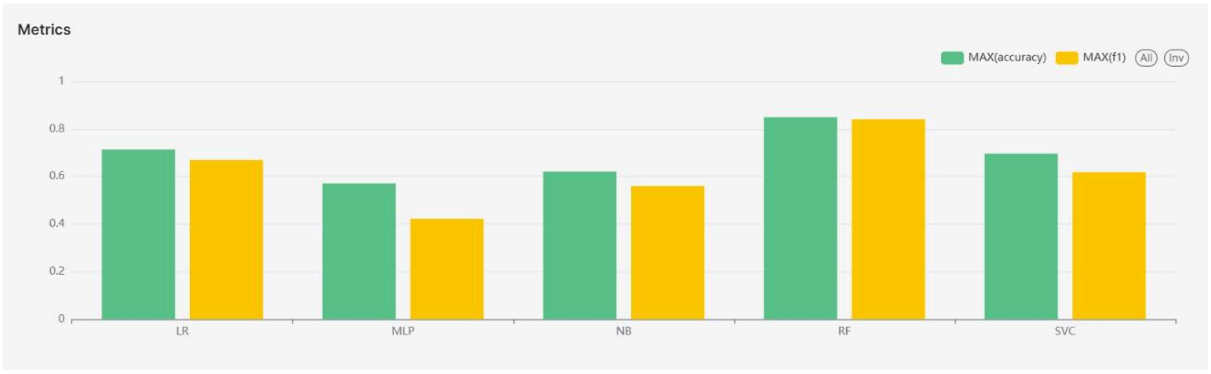
-Models Scores

- **Metrics:** A grouped bar chart comparing Accuracy and F1-score of all five classifiers (LR, MLP, NB, RF, SVC). Random Forest stands out as the top performer with accuracy and F1 around 85%.
- **lr/mlp/nb/rg/svc predictions:** Show actual vs. predicted irradiance levels for each model, a prediction results for few data samples.
- **lr/mlp/nb/rg/svc Confusion Matrix:** These matrices take a look at class-wise performance, helping identify missclasification trends, as for example with MLP and Naive Bayes tending to predict 0 most of the times.
- **Samples:** List example of transformed features used during training and inference, illustrating the result of preprocessing pipeline.

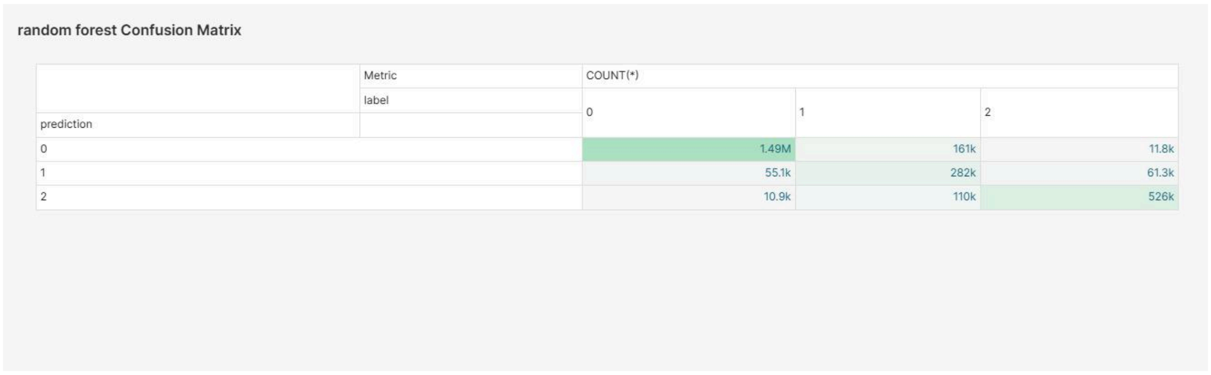
Through data analysis, we got several findings. The records are distributed evenly between the sations, ensuring a balanced dataset across geographic regions.



particularly interesting was the correlation between temperature and level of radiation.



For the ML part, forest happens to be the most effective model, achieving accuracy of 85% and F1-score of 84%, indicating good results across all classes. SVC and LR performed not as good, but still were able to find some correlations in the data and achieve accuracy of about 70%.



random forest Confusion Matrix

	Metric	COUNT(*)		
	label	0	1	2
prediction				
0		1.49M	161k	11.8k
1		55.1k	282k	61.3k
2		10.9k	110k	526k

Confusion matrix clearly shows what all model good at predicting majority class(0), but gets confused on other classes.

8. Conclusion and Reflection

In conclusion, this project successfully demonstrated the potential of leveraging machine learning models for predicting critical weather conditions, specifically solar irradiance. Through rigorous feature engineering and analysis, and the exploration of various modeling approaches on a substantial dataset, we validated our initial hypothesis regarding the feasibility of this concept. While the achieved performance metrics are encouraging and affirm the viability of utilizing ML for this purpose, they also highlight areas for further refinement. Future work should prioritize the incorporation of additional relevant features and a broader expansion of the dataset, both in terms of numerical scale and geographical coverage beyond the initial focus on the USA and Canada, to enhance the model's accuracy and ensure greater generalizability.

The primary challenges encountered during the project stemmed from the process of learning and implementing various frameworks and tools.

- **Python Version and Dependency Issues:** Significant difficulties arose due to Python version incompatibilities and complex library dependencies. A notable example was the inability to import the `datasets` library on Python versions 3.10, 3.11, and 3.12, which was traced back to an incorrect Python installation that the team could not directly resolve.
- **HIVE Data Import Distortion:** Importing data into HIVE presented a serious problem, as the values became significantly distorted after the import process. This distortion negatively impacted subsequent data analysis and the results of model training.
- **Horovod Installation and Configuration:** Additional complexities were faced during the installation and configuration of the Horovod library, which was required for training Deep Learning models.

A substantial amount of time was dedicated to troubleshooting and resolving issues related to Python versions and their dependencies. Furthermore, significant effort was invested in configuring PySpark to achieve optimal training performance.

Task	Description	Dmitry Kuzmin	Yaroslava Bryukhanova	Timofey Brayko	Artemii Miasoedov	Artur Rakhmetov	Deliverables	Average hours spent
Data loading and preprocessing	Load dataset from Hugging face. Get rid of redundant columns. Save dataset in CSV files.	50%	0%	50%	0%	0%	stations.csv records.csv	4
SQL ingestion	Prepare SQL tables and ingest prepared dataset.	0%	0%	100%	0%	0%	records.parquet stations.parquet	12
Hive ingestion	Creating and optimizing tables	10%	90%	0%	0%	0%	records.parquet stations.parquet	12
EDA	Writing insight queries, Adding info to Superset	80%	20%	0%	0%	0%	qx.csv/qx.jpg	10
Feature Engineering	Prepare feature extraction pipeline using Pyspark engine	0%	0%	0%	50%	50%	train.json test.json	12
ML modeling	Train models	0%	0%	0%	70%	30%		18
Deep Learning	train deep model	0%	0%	0%	100%	0%	output/model*.cs v	14
Dashboard	visualization of data	50%	0%	0%	0%	50%		8
Report & Presentation	Prepare presentation and report of done work	20%	20%	20%	20%	20%	Report.pdf Presentation.pdf	3