Redback Operations
Data Science and Analytics Team

Documentation

Project 5 - Workout Analysis and Data Modelling

Project Overview

Whether a bike user is using an indoor trainer, cycling outdoors, or using Redback Operation's VR product, being able to provide post-workout analysis and succinct data visualisations to the user is a critical component of the user experience. Being able to develop analytical models and tools to analyse workouts will also benefit Redback Operation's ability to create popular/in-demand structured workouts. Furthermore, by beginning to analyse user data, Redback Operations can begin ranking users thus creating a competitive in-game environment.

Key Aims Trimester 3 2022

Aims for Trimester

1. Initiate project (scoping) documentation capturing key technical considerations, research, deliverables etc.
2. Conduct competitor research regarding visualisation and post-workout analysis features and standards.
3. Procure a meaningful amount of workout data to support model analysis and training efforts.
4. Uniformly organise and clean procured data into respective datasets and a single, easily accessible database.

5. Draft a plan to integrate data analysis efforts with Projects 2 and 3 i.e., included calculators and oxygen prediction model within the data visualisations.
6. Develop at least one prediction-based model i.e., Workout category prediction (climbing, HIIT, recovery etc).
7. Begin developing Tableau-based reporting visualisations.

Deliverables

1. Thorough Project documentation.
2. Thoroughly conduct competitor research specific to the project scope.
3. Organised data location of procured data.
4. Prototype of data visualisation within Tableau concerning post-workout analysis.
5. Deployment (and testing) of one predictive learning model.
6. Thoroughly document key handover materials and upload them to the DSA GitHub repo.

Key Outcomes (Summary)

1. Thorough documentation was developed and published concerning research and development efforts
2. Large data acquisition effort - Over 4500 hours of exercise data was acquired (manually), converted into a readable format, cleansed and organised into respective data tables.
3. A Bigquery environment was set up to house all critical datasets
4. SQL scripting was used to further refine and process the data in anticipation for modeling efforts.
5. A prototype dashboard was developed, however in the interest of time and technology simplicity (Google-centric), Google Looker Studio was used instead of Tableau.
6. Three different models were used to predict Power (wattage) output during workout sessions.
7. All documentation was uploaded to the DSA Github repository.

Project Approach

The project was split up into 5 phases; 3 deliverable phases and 2 non-deliverable phases:

Non-deliverable:

1. Research
2. Upskills

Deliverable:

1. Data Handling and Analysis
2. Modeling

3. Publish/Documentation

1/ Research

Once the project scope was defined, planning and research tasks materialised. The Machine Learning Workflow ^^ was used to structure up efforts:

1. Required Skills - Identify gaps
2. Retrieve Data
3. Clean and explore
4. Prepare/Transform
5. Develop and Train the Model
6. Validate and Evaluate
7. Deploy

Required Skills

After reviewing a number of articles concerning Data Science and Machine Learning workflows, it was clear that the following skills were going to be critical in navigating this project:

The skills were colour coded to provide a view of what skills required ‘upskilling’ (red), which skills were at an acceptable level (green) or which skills were developed enough to navigate the requirements of the project but could be improved.

Technical Skills:

1. Cloud computing - Google Cloud
2. Data visualization tools - Google Looker Studio
3. Query languages - SQL
4. Database management - BigQuery
5. Programming/Coding - Python
6. Microsoft Excel - CSV Files and Data handling
7. Data wrangling - Pandas / Python

Broad Skills:

1. Statistics and probability
2. Advanced Mathematics
3. Machine learning

On review, python, google cloud and skills needed to be advanced (upskilled) in order to successfully navigate this project; Unfortunately, the units covered through Trimesters 1 and 2 of the Bachelor of Artificial Intelligence have not covered Python or Google Cloud-related

skills/material. As result, considerable effort was invested in upskills in the areas highlighted in red.

Retrieve Data aka 'Data Acquisition

Whilst there are many datasets available online, the approach to this project was to build it from the ground up and not 'skip' any elements or stages in the workflow. As a result, a data acquisition requirement materialised along with a range of questions; Where can data be acquired from, what format will the data come in, and how can the acquired data be formatted in preparation for data cleaning and organisation efforts?

The following resources were used:

- What devices are used to record cycling workouts - https://www.cyclingnews.com/features/best-cycling-computers/
- How to pull data from a bike computer https://www.android.com/filetransfer/ https://www.android.com/filetransfer/
- How to connect a device to a computer - https://support.wahoofitness.com/hc/en-us/articles/115000127910-Connecting-ELEMNT-BOLT-ROAM-to-Desktop-or-Laptop-Computers
- What format are transferred https://support.wahoofitness.com/hc/en-us/articles/115000127910-Connecting-ELEMNT-BOLT-ROAM-to-Desktop-or-Laptop-Computers workout files in -
- What is a fit file - https://developer.garmin.com/fit/protocol/
- Fit file information - https://developer.garmin.com/fit/overview/
- How to convert a fit file to a cv file and resources:
    - https://developer.garmin.com/fit/overview/
    - https://www.youtube.com/watch?v=MwN-NrkjSUY&ab_channel=franchyze923
    - https://www.youtube.com/watch?v=k8vrTdcHtt0&ab_channel=RDGames%26Tech
    - https://github.com/rdchip/FIT-to-CSV-converter-for-windows
    - https://www.python.org/downloads/release/python-3110/
    - https://www.jetbrains.com/edu-products/download/#section=pycharm-edu
    - https://pypi.org/project/fitparse/#description
    - https://github.com/dtcooper/python-fitparse
    - https://github.com/ekapope/Combine-CSV-files-in-the-folder/blob/master/Combine_CSVs.py

Data Cleansing and Organisation

Once the workout files were able to be converted into CSV format, determining the best way to organise the workout files into a structured format was critical; in preparation for effective data storage and organisation purposes. As a result a number of questions surfaced; how can I automate or semi-automate the process (there were hundreds of workout files per user)? How can I combine a respective User's work files into one file and how can I automate this?

The following resources were used:

- https://www.geeksforgeeks.org/rename-multiple-files-using-python/
- https://towardsdatascience.com/how-to-merge-large-csv-files-into-a-single-file-with-python-c66696f595ff
- https://gist.github.com/anbento0490/b17b08fccd246d1ff95c9cbba6c38b19#file-merge_csv_files_1-py

Following the efforts to get FIT files into organised CSV files - determining how to store data, and add information to datasets by way of SQL scripts, clean data, create data tables, create storage locations (for large CSV files), query the database and create tables from the results, automate data transfers were all assessed using the below resources:

- https://cloud.google.com/bigquery/docs/how-to
- https://cloud.google.com/bigquery/docs/introduction
- https://cloud.google.com/bigquery/docs/quickstarts/query-public-dataset-console
- https://cloud.google.com/bigquery/docs/quickstarts/load-data-console
- https://cloud.google.com/bigquery/pricing
- https://cloud.google.com/storage/docs
- https://cloud.google.com/bigquery/docs/batch-loading-data
- https://cloud.google.com/bigquery/docs/reference/standard-sql/introduction
- https://cloud.google.com/bigquery/docs/dts-introduction
- https://cloud.google.com/bigquery/docs/exporting-data
- https://hevodata.com/learn/google-bigquery-sql/
- https://github.com/cathytanimura/sql_book

Dashboard/Reporting

Once Data was available in the Bigquery Database, the ability to build a Dashboard report was then available. The following resources were used:

- https://cloud.google.com/bigquery/docs/visualize-looker-studio
- https://blog.hubspot.com/marketing/google-data-studio
- https://support.google.com/looker-studio/answer/6309867#zippy=%2Cin-this-article

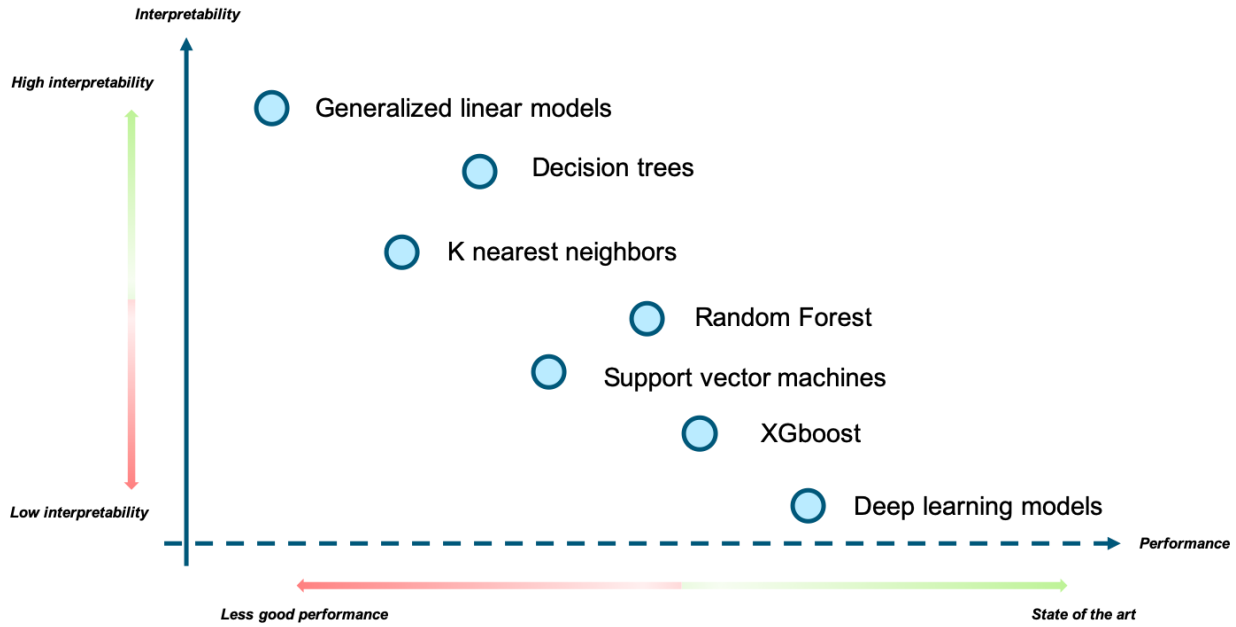
Modeling

Initially, research was conducted focusing on the best model fit for the problem (the target is power output prediction). A number of resources were used to help inform model selection:

- Interpretable vs. noninterpretable machine learning models for data-driven hydro-climatological process modeling - https://www.sciencedirect.com/science/article/pii/S0957417420311428
- The balance: Accuracy vs. Interpretability - https://towardsdatascience.com/the-balance-accuracy-vs-interpretability-1b3861408062
- Interpretable vs Explainable Machine Learning - https://towardsdatascience.com/interperable-vs-explainable-machine-learning-1fa525e12f48

- Considerations when choosing a machine learning model - https://towardsdatascience.com/considerations-when-choosing-a-machine-learning-model-aa31f52c27f3
- An Easy Guide to Choose the Right Machine Learning Algorithm - https://www.kdnuggets.com/2020/05/guide-choose-right-machine-learning-algorithm.html
- Interpretability versus explainability - https://docs.aws.amazon.com/whitepapers/latest/model-explainability-aws-ai-ml/interpretability-versus-explainability.html

The outcome of the research is summarised in the below diagram. Two interpretable and supervised models were selected - Generalised linear models and Random Forest.



2/ Data Acquisition, Data Cleansing and Organisation

Redback Operations
Data Science and Analytics Team
M.TELLEY

DOCUMENTATION
Data Procurement, Cleansing and Organisation

Purpose

Redback Operation core product will collect performance (output) data from a number of sensors while the user is using the bike and wahoo kickr trainer. Moreover, establishing a working environment that will align with the core product is critical in establishing continuity in data-related tasks and advising other core teams about Data Science and Analytics (DSA) Team's requirements. The initial effort is to set up a core data location for the team that houses all key datasets.

By the end of this task, further work on data analysis, visualisation and using a number of machine learning algorithms will be possible.

Scope

Initialise a BigQuery (Google) environment that houses all of the DSA's data; three key datasets have been identified:

1. Sales device data
2. Oxygen (O2) Uptake-related data
3. Wahoo/Garmin workout data.

Sales and O2 datasets were located within the DSA team's GitHub repository, however, the Garmin and Wahoo data was procured, cleansed and organised independently. This document outlines the specific details relating to independent data procurement.

Wahoo and Garmin fitness activity data procurement

What is / Who are Garmin and Wahoo?

Wahoo
Founded in 2009 by Chip Hawkins in Atlanta, GA, Wahoo creates innovative solutions to make hard-fought goals attainable and lives better. Wahoo was built on the foundation of simplicity and the mindset that "there's got to be a better way. Moreover, Wahoo produces a suite of devices that record/collect performance-related data from Athletes.

Key devices involved in the data procurement process:

1. Wahoo Bolt
2. Wahoo Elemnt

Garmin
Gramin makes products that are engineered on the inside for life on the outside. We do this so our customers can make the most of the time they spend pursuing their passions. Garmin brings GPS navigation and wearable technology to the automotive, aviation, marine, outdoor and fitness markets.

Key devices involved in the data procurement process:

1. Edge 1030
2. Edge 1040

File output - FIT files

The Flexible and Interoperable Data Transfer (FIT) protocol is designed specifically for the storing and sharing of data that originates from sport, fitness and health devices. The FIT protocol defines a set of data storage templates (FIT messages) that can be used to store information such as activity data, courses, and workouts. It is designed to be compact, interoperable and extensible. Extensive documentation is available here regarding FIT files.

The FIT file protocol was designed to provide:

- Interoperability of device data across various platforms
- Scalability from small embedded devices to cloud platforms
- Forward compatibility, allowing the protocol to grow and retain existing functionality
- Automated compatibility across platforms of different native endianness

Referenced from *Garmin*

FIT files are not immediately in a format that a human can read such that a conversion to a Comma-separated values file (CSV) is required. This document outlines the steps required to conduct a CSV conversion.

Where was the data procured from?

I provided over 400 FIT files to the project and also procured data from our Cyclists in Melbourne by way of convenience sampling. Therefore, the activity files can't be used for population sampling activity; demographics are extremely narrow i.e., Male, 30-60 years etc.

Accessing the data

Computer used to pull FIT file: Macbook Pro (iOS)

Garmin Devices: A USB-c cable was used to access the files on the device. The device is immediately accessible in the device listing; files are then transferred (copied) off the device.

Wahoo: To access a Wahoo device, Android File Transfer is required. The device also needs to be turned ON. Thereafter, the device is immediately accessible in the device listing; files are then transferred (copied) off the device. More information is available here about connecting a Wahoo device to a computer.

Other information collected

In addition to acquiring FIT files from a participant, their age, biological gender (male/female), weight and functional threshold power (FTP) were also recorded.

Privacy

FIT files contain GPS data and device data such as serial numbers; these data points have been considered sensitive and have been removed from the datasets entirely. Moreover, participants signed a statement of release which outlined what the data was going to be used for and how the data was going to be anonymised.

FIT File Conversion, Data cleansing and Organisation

FIT Conversion

Initially, Garmin Development documentation was assessed to understand the basics. Garmin's FIT software development kit (SDK) was installed and the FitCSVTool was used to convert FIT

files to CSV format. For a demonstration, please watch this video. Note: the FIT SDK was downloaded and used on a Computer running a Windows operating system.

Testing was conducted initially using a Wahoo-derived FIT file. On review, the output wasn't organised in a way that instantly supported data analysis-related tasks. Further research was conducted to refine the conversion process.

A key resource used established a pathway to securing improved conversion techniques using a python script to convert FIT files in a highly organised manner. Moreover, the python script for wahoo-derived FIT files also included two additional functions that renamed the files to secure standardised naming conventions, removed sensitive data fields and also combined all of the CSV files into one master CSV file i.e, one master file per participant.
Key resources:

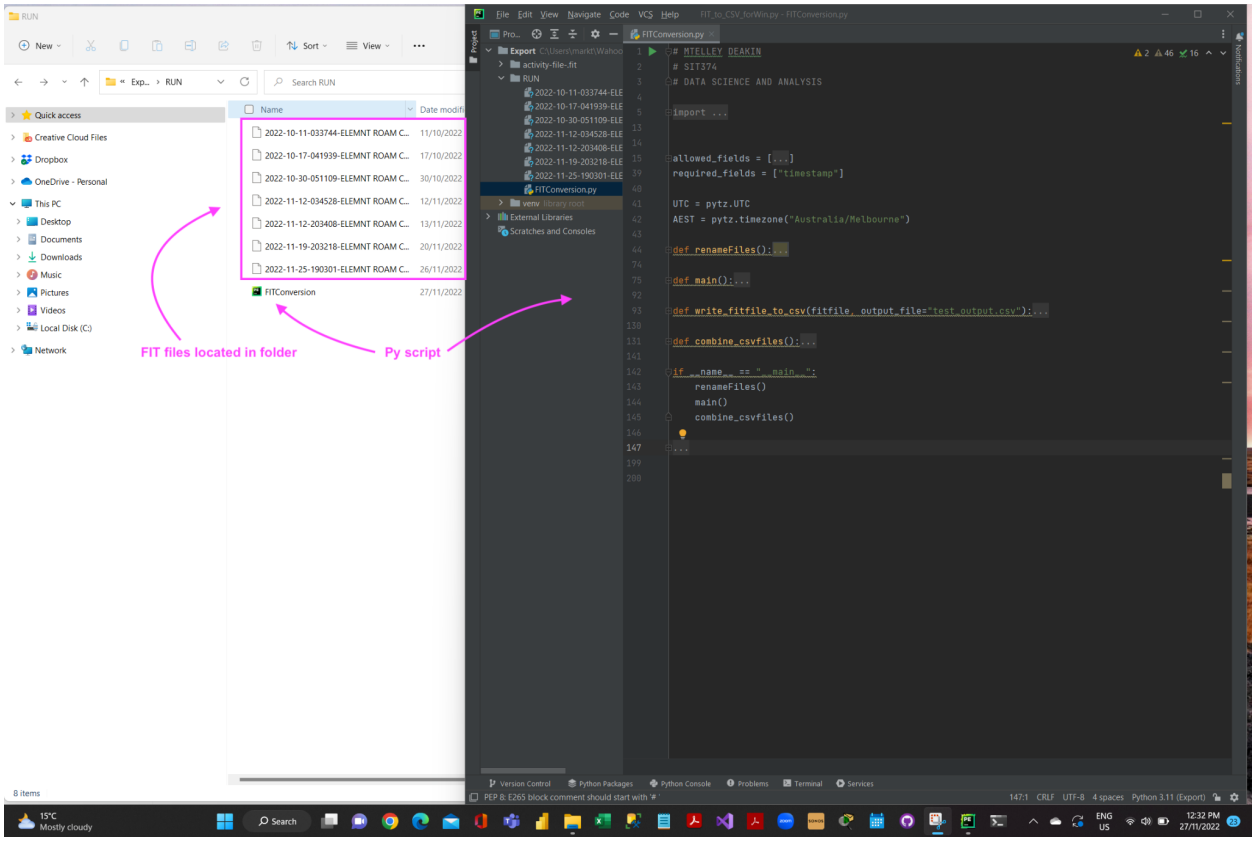
- https://github.com/rdchip/FIT-to-CSV-converter-for-windows
- https://www.python.org/downloads/release/python-3110/
- https://www.jetbrains.com/edu-products/download/#section=pycharm-edu
- https://pypi.org/project/fitparse/#description
- https://github.com/dtcooper/python-fitparse
- https://github.com/ekapope/Combine-CSV-files-in-the-folder/blob/master/Combine_CSVs.py

Data Cleansing and Organisation

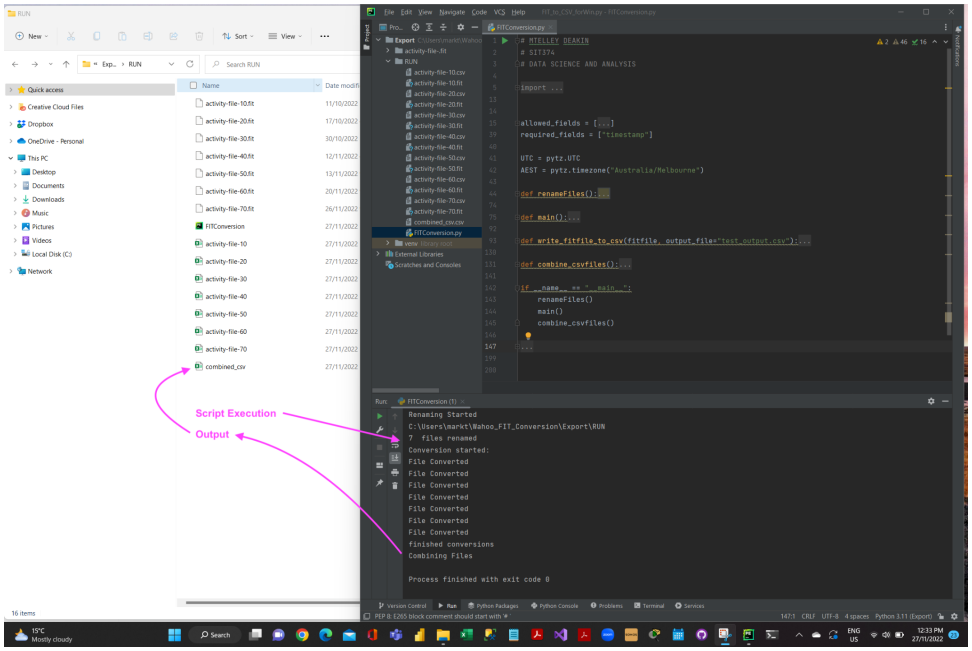
Given the scale of data being procured, BigQuery (Google) was used to house and organise the data. Due to the size of the user CSV files, they were stored in a Google Cloud Storage bucket in preparation for housing the data in a data table. On review, the schema of the CSV file relating to data types also presented challenges; all data types were set as STRING values initially to avoid data table creation errors. The schema file used is located within the Github repo.

Key Operations (Wahoo)

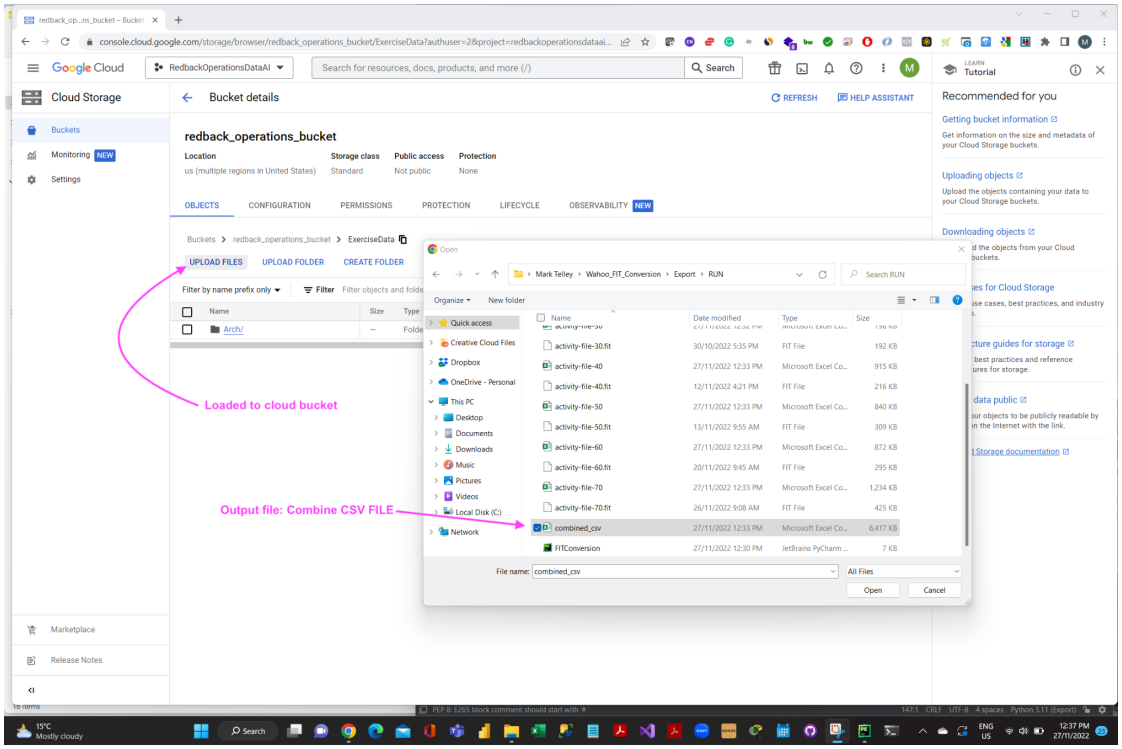
Open the script file, drop FIT files into the correct folder and run.



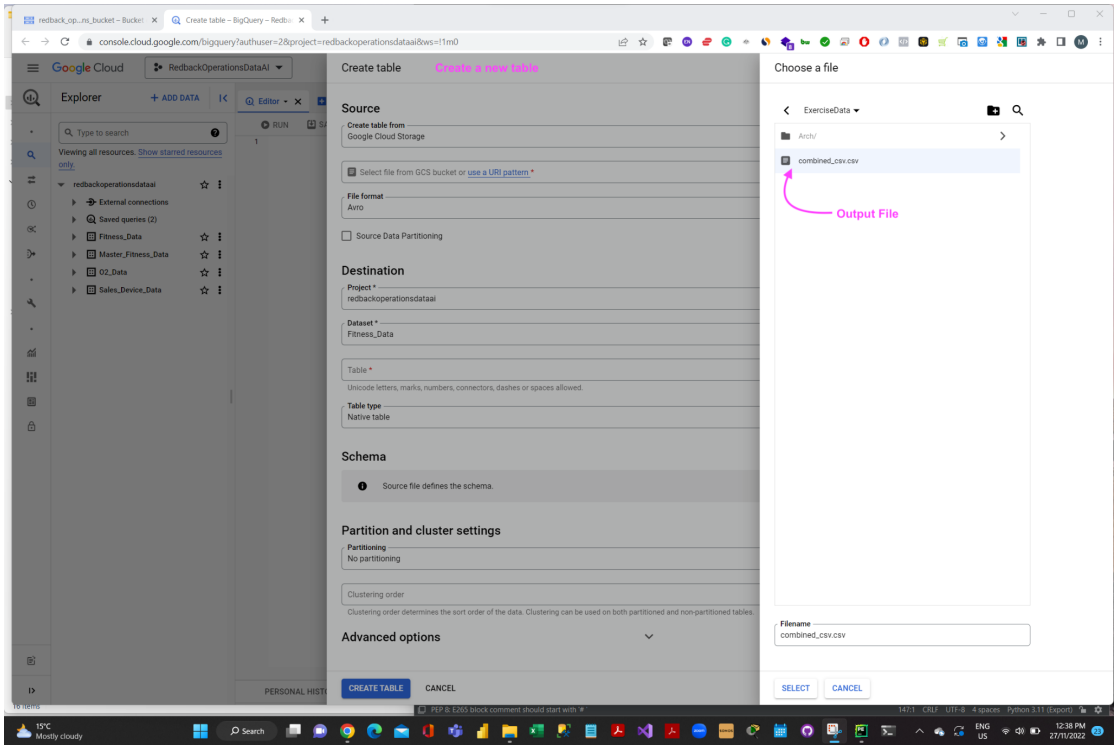
After the script has finished, a combined CSV file will be available in the same location + all single FIT files have been renamed and converted. Upload the combined CSV file to a Google Cloud storage bucket:



Upload the CSV file to a Google Cloud Storage bucket:

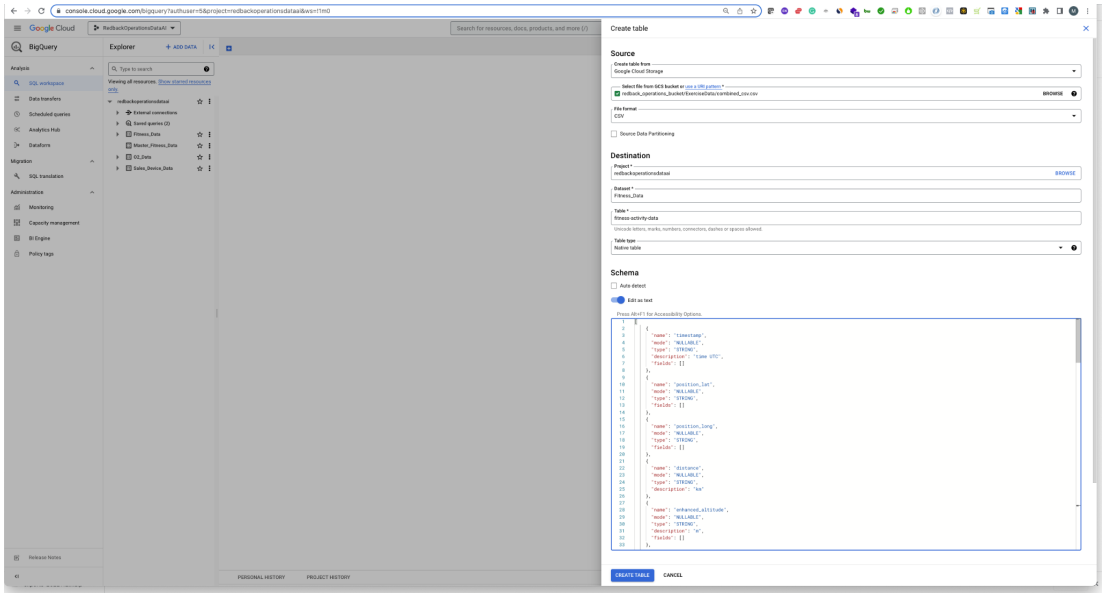


Create a data table: Load from the cloud storage bucket:

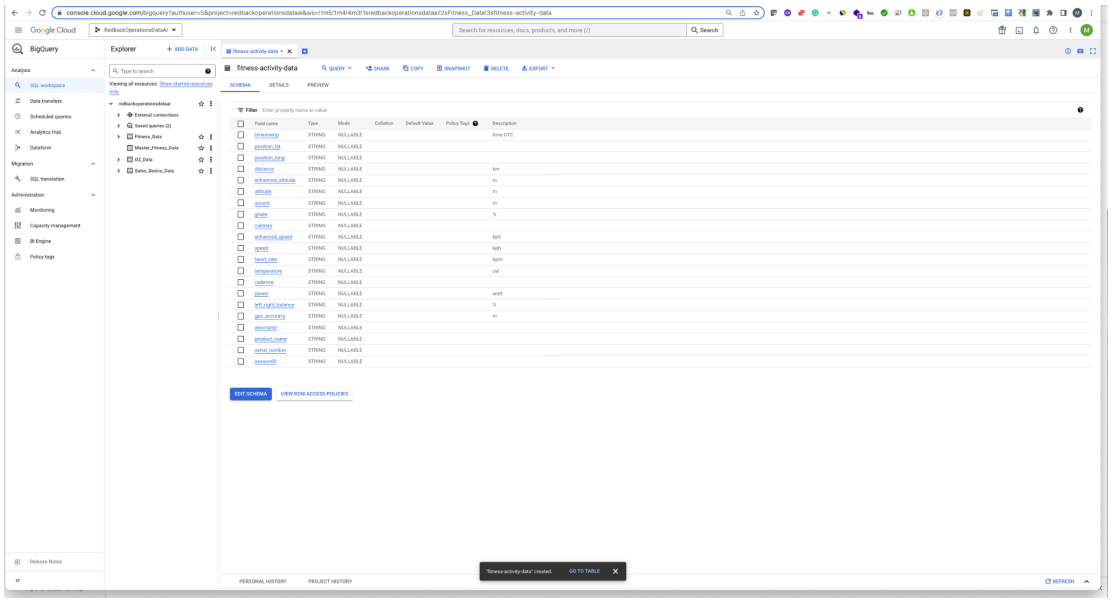


When creating a data table, local files up to 100MB can only be loaded, hence the need to store them in a Cloud bucket.

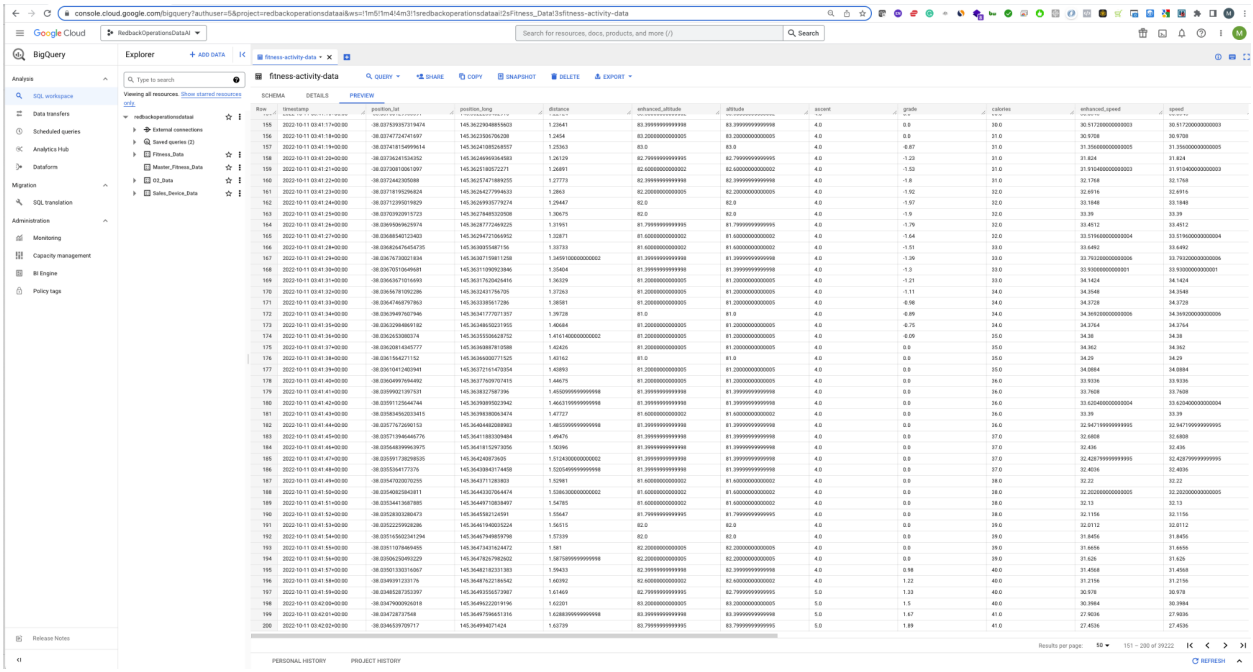
Copy in the table schema JSON code:



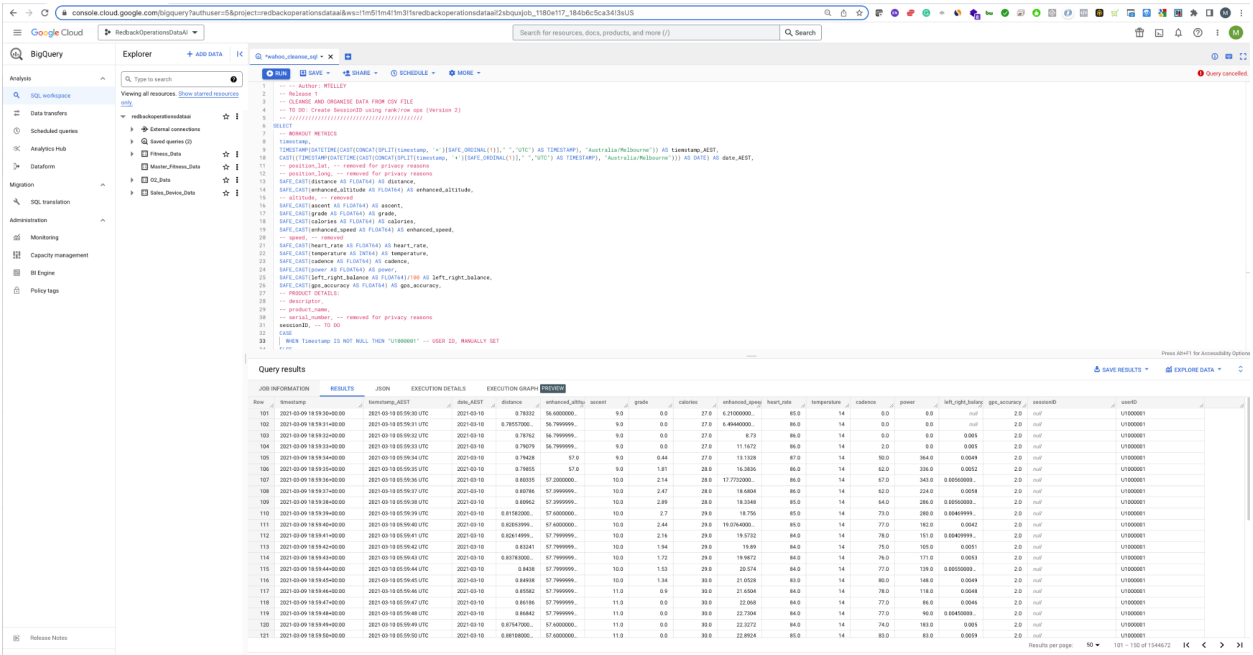
The table has been created, with singular data types:



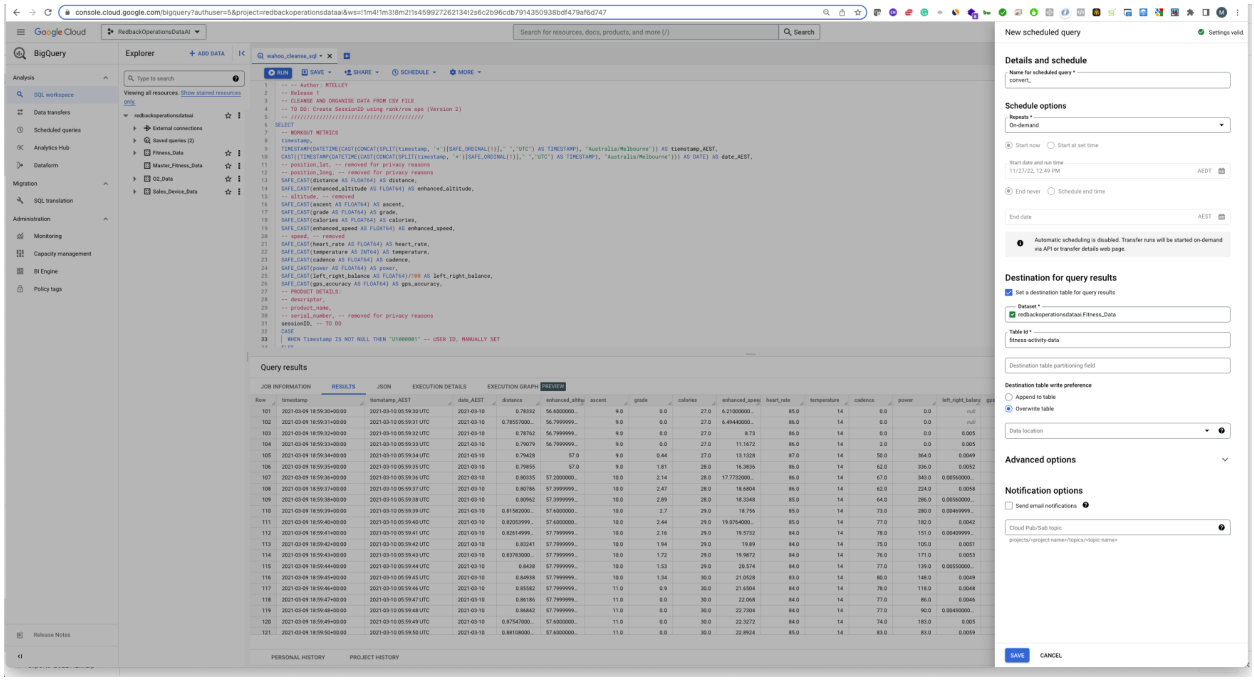
Preview of data:



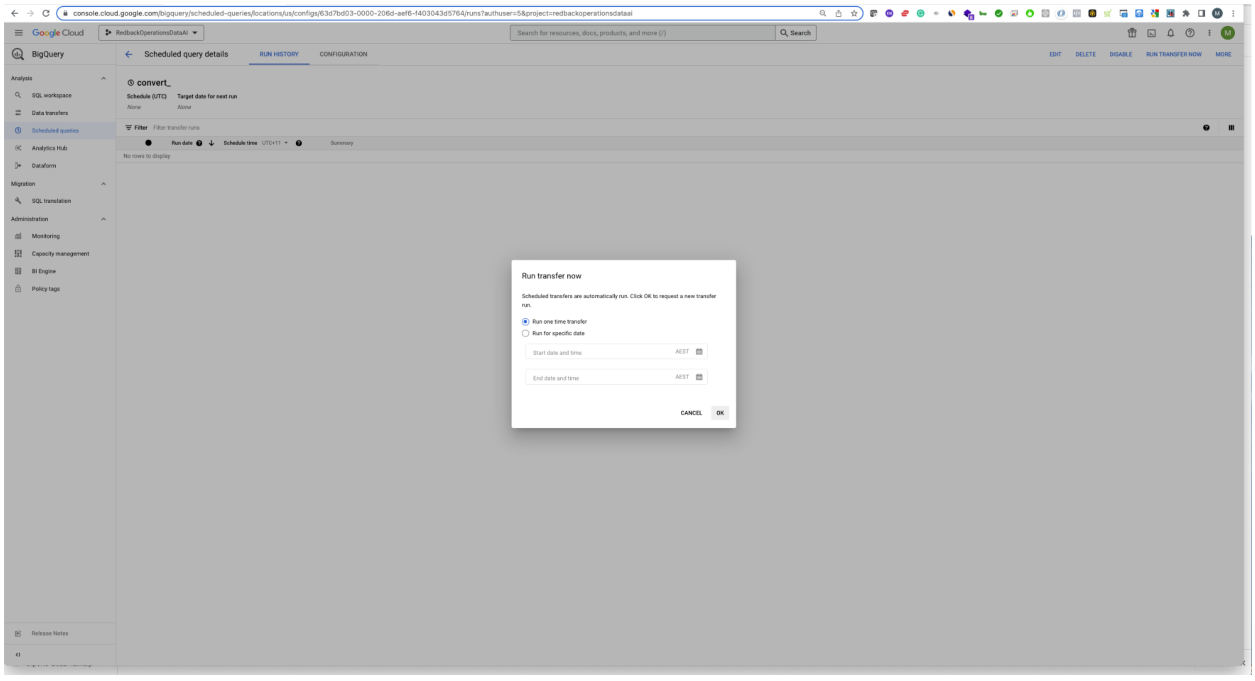
Run script to clean up the data table; The script will correct data types, key user data manually added, sensitive data removed and some filtering logic applied:



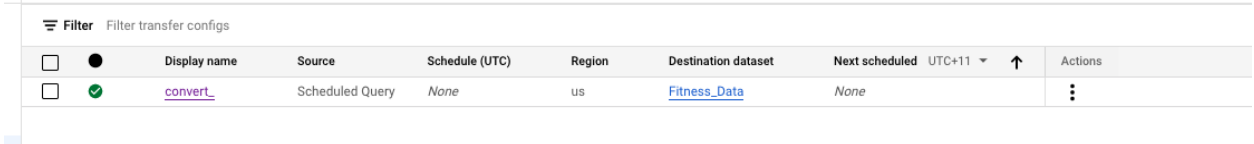
Schedule the query to create (overwrite) a new table



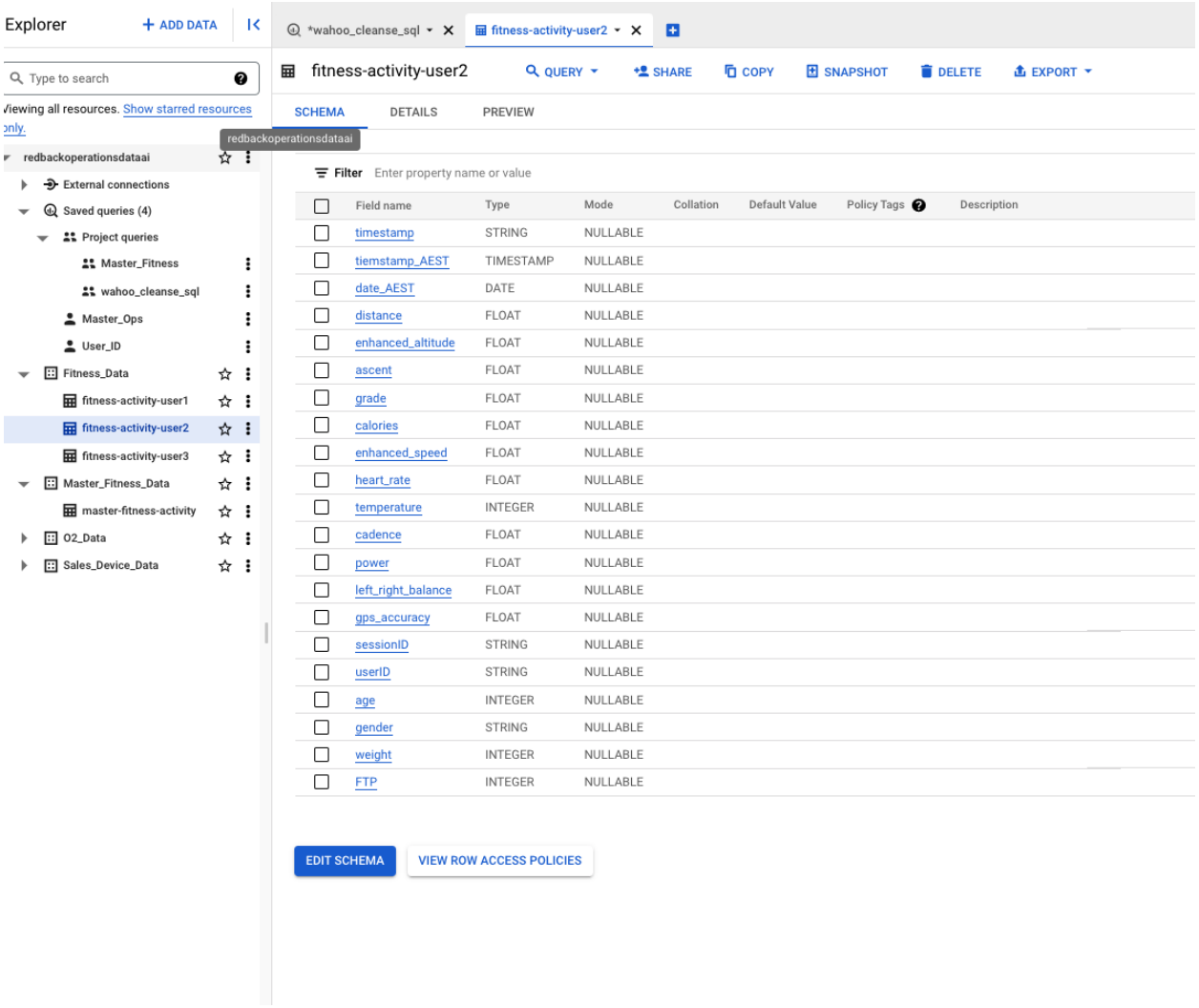
Then run the transfer:



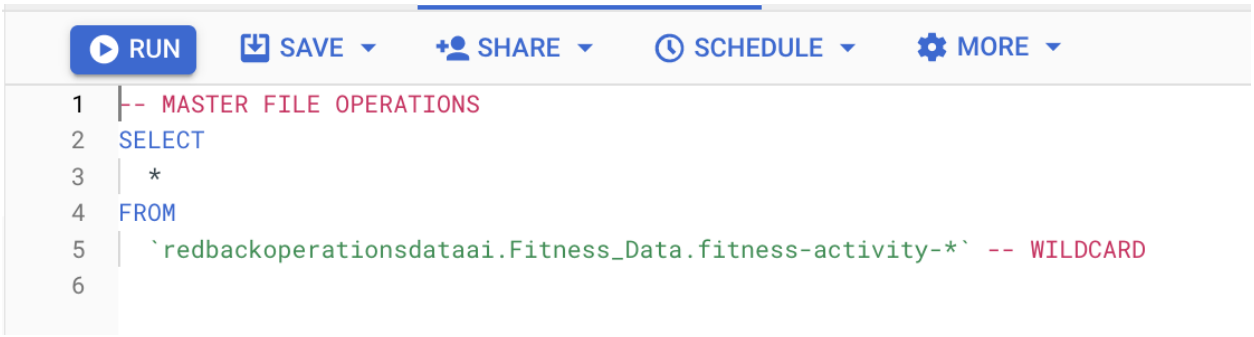
When complete, the green tick will indicate the transfer is complete:



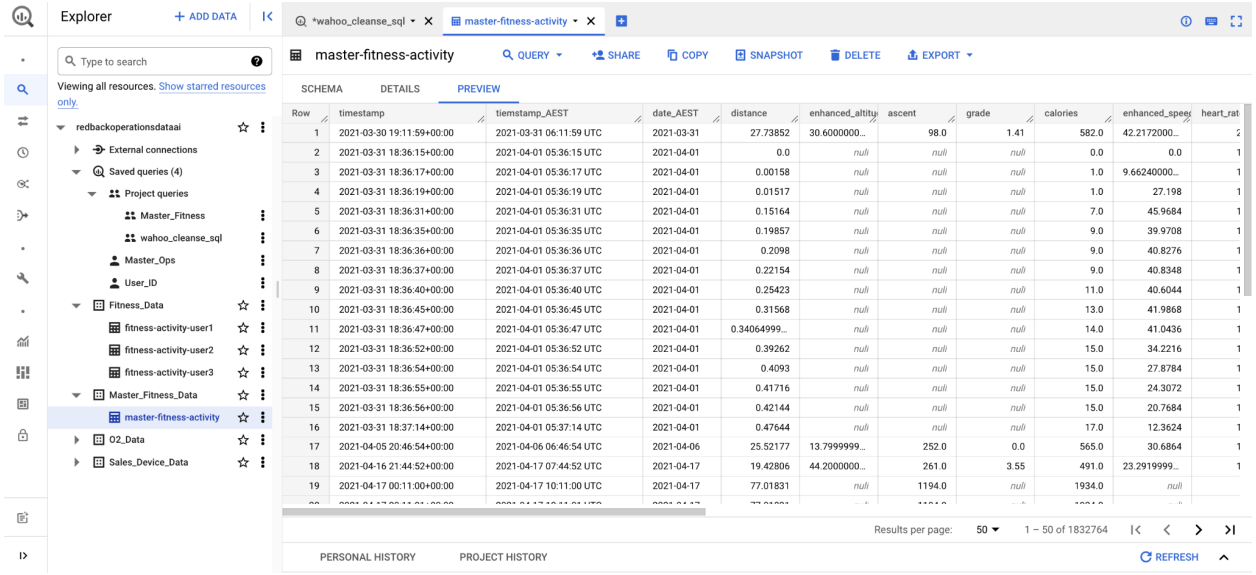
Updated Schema (new data types etc):



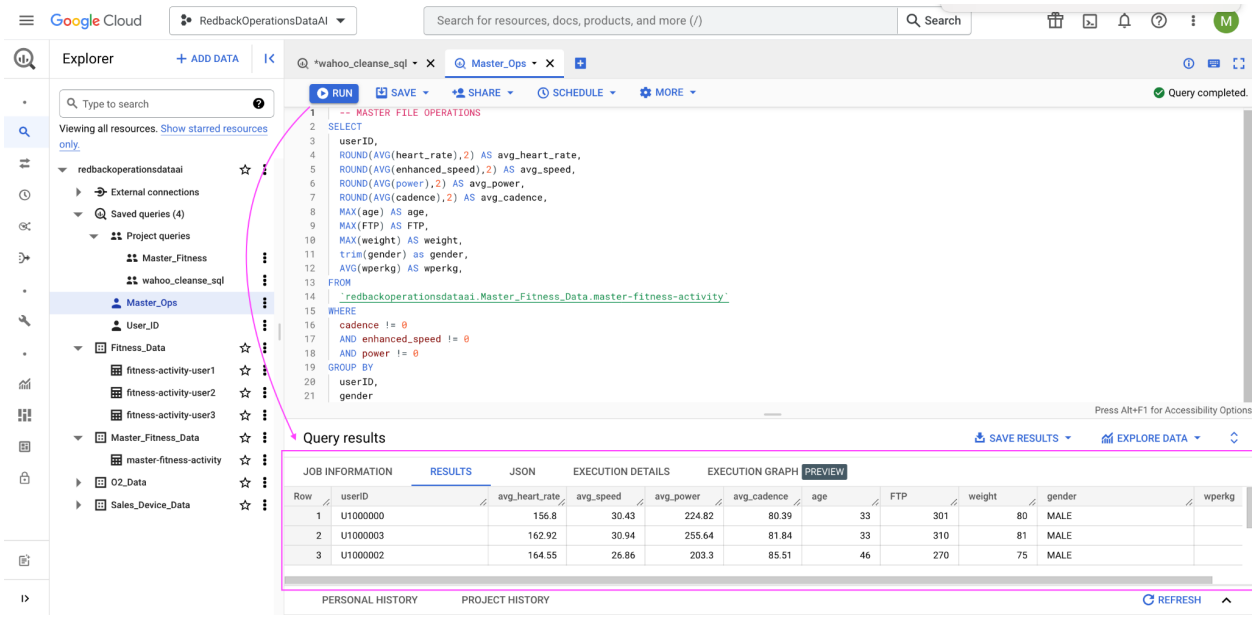
The following query is then scheduled: Given the consistent naming conventions, a wildcard operation can be used to join user tables together:



The mater table is created which houses all user data:



This master table will be used as a key resource; The table can be queried, connected to Tableau etc for data analysis and modeling efforts:



The process was replicated for additional users.

Key Operations (Garmin)

There was a slight change in the set of operations for Garmin-derived data. As a result, certain attributes were omitted i.e., Left/Right balance. To simplify operations and in the interest of time, only one user with Garmin-derived data was included in the final data set. Further work is required to support and update key operations for Garmin-derived data.

Access to Google Console Assets

Google Console Storage

Access to Google Cloud Storage and the relevant project buckets has been restricted due to data privacy requirements and billing controls; I'm using my own personal Google Cloud Storage account to store combined CSV files.

Google Bigquery

Relevant project team members have had access enabled for all datasets - Team members only have viewing privileges. Access details below:

Admin/Owner: mtelley@deakin.edu.au

Access information - watch this Loom video.

Other Notes

Please note, there is availability to query a data table when saving the query results to another data table. This method was not used for a range of reasons, mainly due to the transfer method providing added debugging capabilities should errors occur.

There is an outstanding task that requires some minor SQL development to allocate session IDs to workouts i.e, there can be more than one session in a day, such that each session must have a unique ID. This will be achieved using lag/over operations:

```sql
WITH
  proto_1 AS (
  SELECT
  timestamp,
    LAG(Distance) OVER(ORDER BY timestamp ASC ) AS PreviousRow,
  CASE
      WHEN Distance<LAG(Distance) OVER(ORDER BY timestamp ASC ) THEN 1
    ELSE
    0
  END
    AS endSession,
  FROM
    `redbackoperationsdataai.Fitness_Data.fitness-activity-user1`
  ORDER BY
    Timestamp ASC)
SELECT
  *,
FROM
  proto_1
WHERE endSession != 0
```

Query results

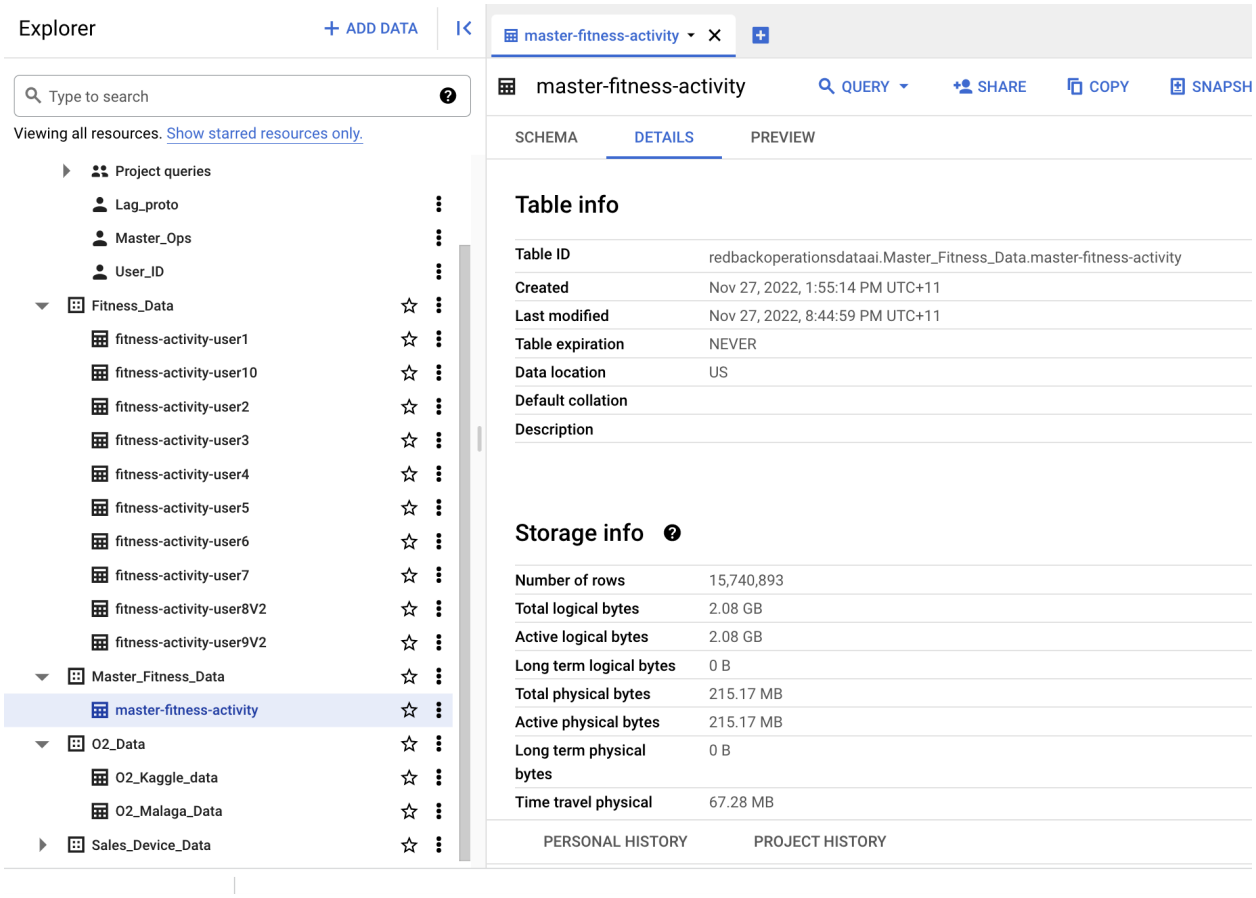
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	timestamp	PreviousRow	endSession
1	2021-03-09 22:22:07+00:00	62.928	1
2	2021-03-11 22:43:39+00:00	17.13455	1
3	2021-03-13 19:53:22+00:00	10.30559	1
4	2021-03-14 20:54:21+00:00	73.90983	1
5	2021-03-15 07:33:59+00:00	5.18281000…	1

Outcome

Over 15,740,893 seconds (+4300 hours) of exercise data has been collated and stored in a single data set. This data set will be used to create data visualisations, train/test models etc. Moreover, the outcome of this exercise will help inform key data collection efforts for the in-game experience i.e., how to collect data, deploy models and then show the users (at the end of their workout or session) a data visualisation that represents their efforts.

Next page for a summary of the master table:



Results

- Successful and well-documented process for converting FIT files into organised CSV files.
- Successful setup of Google Cloud Storage and Bigquery
- After data cleansing; 4,375 hours of the workout were collected
- A number of queries are available concerning data manipulation and pre-processing.
    - Scripts can be found here: <INSERT>
- Datasets for 10 users are available in BigQuery
- Additional information was added to the tables concerning weight, age, userID data etc.

3/ Dashboard

Redback Operations

Data Science and Analytics Team

M.TELLEY

DOCUMENTATION

Dashboard

Live link, Deakin access only:
https://datastudio.google.com/reporting/72bf3538-2b54-4ab7-95a7-c8517cc449e7 \

Purpose

Redback Operation core product will collect performance (output) data from a number of sensors while the user is using the bike and wahoo kickr trainer. Moreover, establishing a working environment that will align with the core product is critical in establishing continuity in data-related tasks and advising other core teams about Data Science and Analytics (DSA) Team's requirements. The initial effort is to set up a core data location for the team that houses all key datasets.

By the end of this task, further work on data analysis and the use of a number of machine-learning algorithms will be possible.

Scope

Once the data is correctly formatted and processed in Bigquery, set up, design and develop a prototype Google Looker Studio dashboard that visualises a User's workout data (by single session). The outcome of this work will further inform future efforts to provide a post-workout analysis to Users once they have finished their session on the bike.

Google Looker Studio

What is Looker Studio?

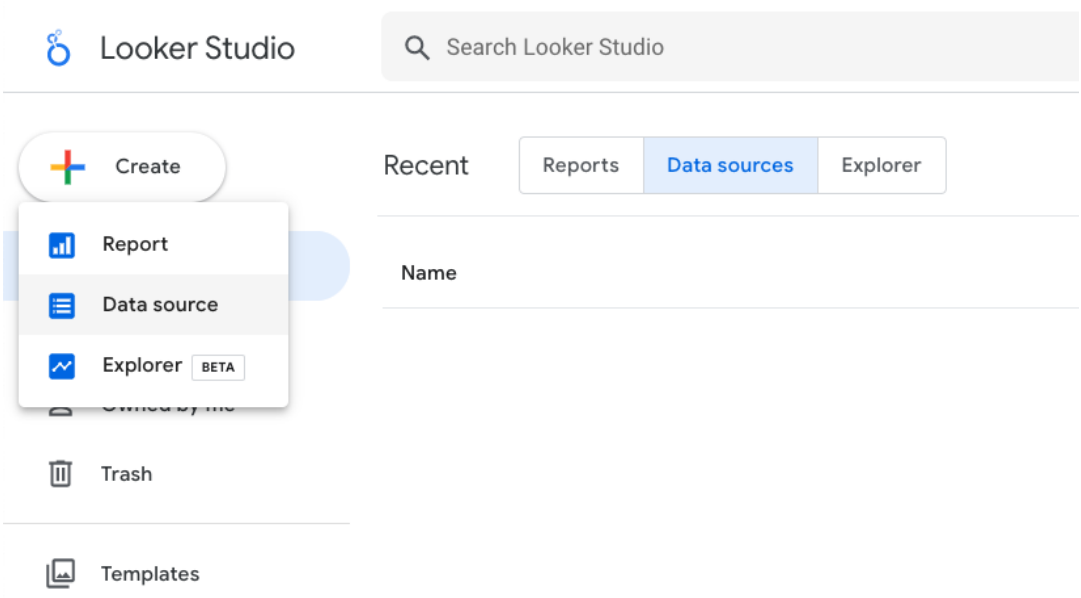
Looker Studio is a free tool that turns data into informative, easy-to-read, easy-to-share, and fully customisable dashboards and reports. Google Looker Studio can:

- Tell a data story with charts, including line, bar, and pie charts, geo maps, area and bubble graphs, paginated data tables, pivot tables, and more.
- Makes reports interactive with viewer filters and date range controls. The data control turns any report into a flexible template report that anyone can use to see their own data.
- Include links and clickable images to create product catalogs, video libraries, and other hyperlinked content.
- Annotate and brand reports with text and images.
- Apply styles and colour themes that make data stories works of data visualisation art.

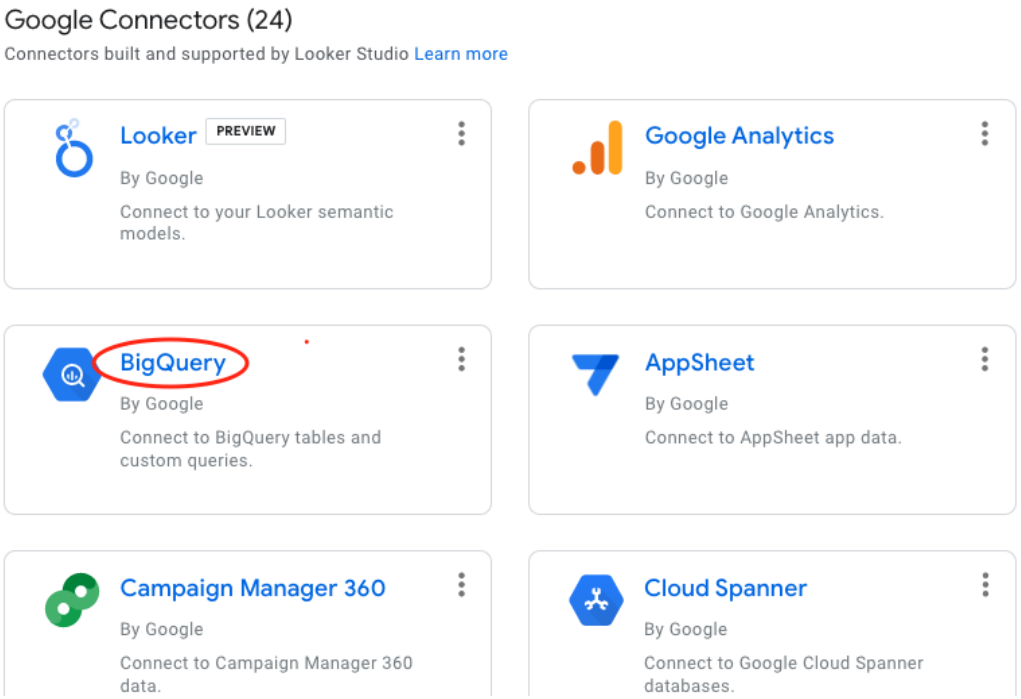
Google Support

How is Looker Studio Used?

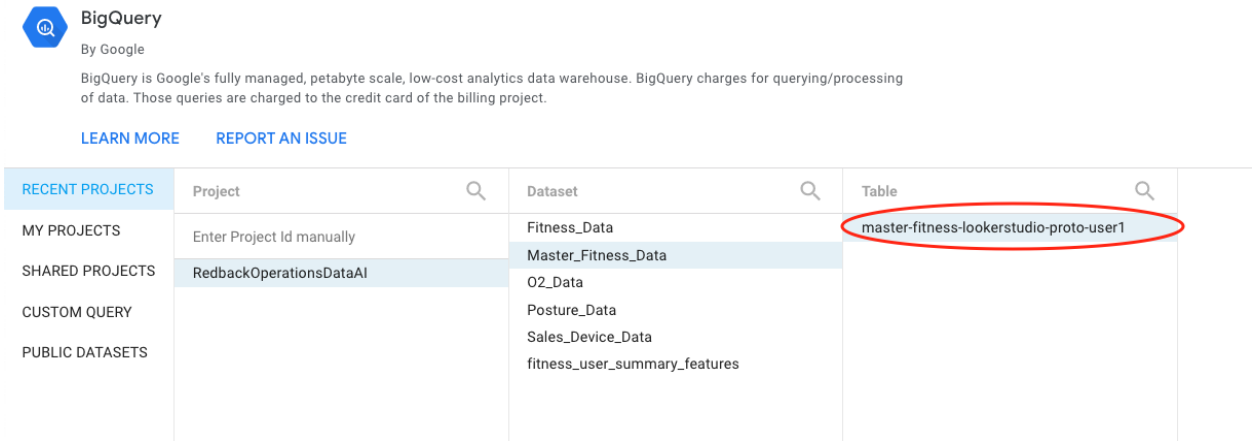
Once the data table is set up in BigQuery, a data source connection is set up in Looker Studio:



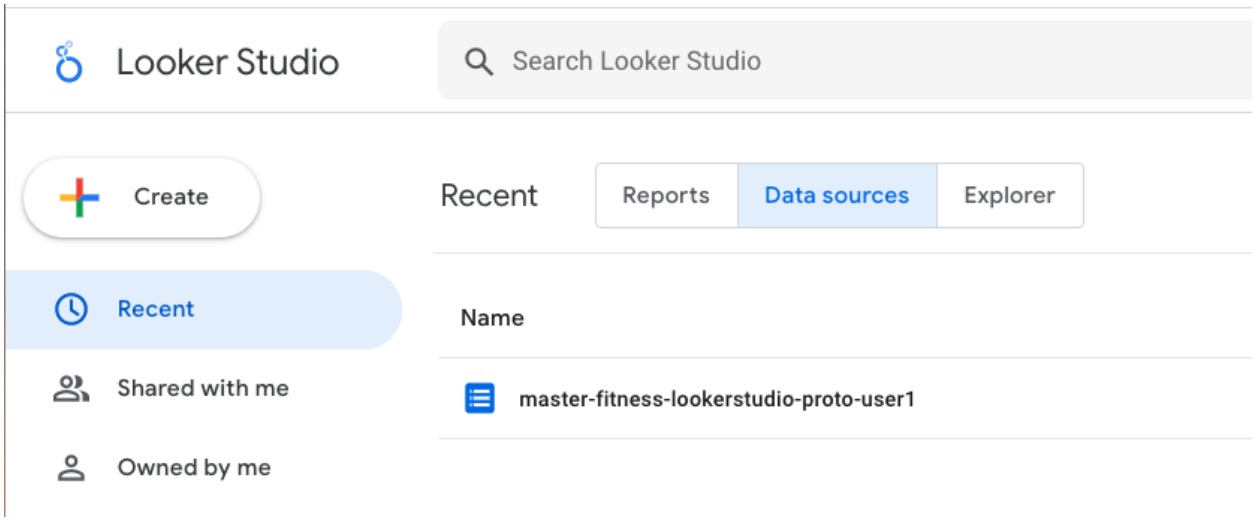
Bigquery is then selected:



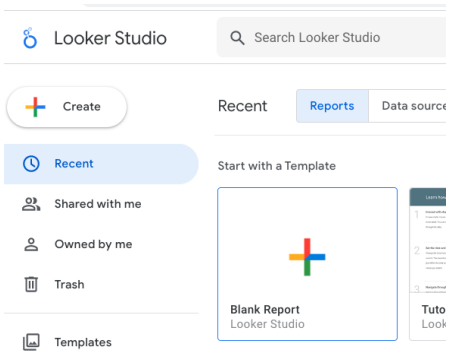
Select the correct data table:



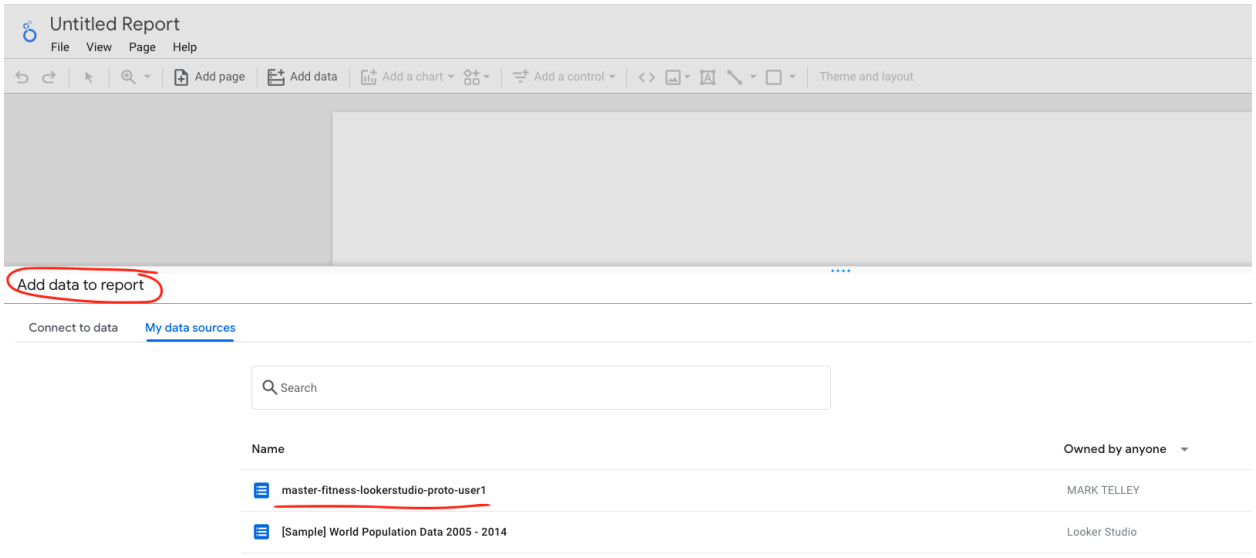
The data source is now available:



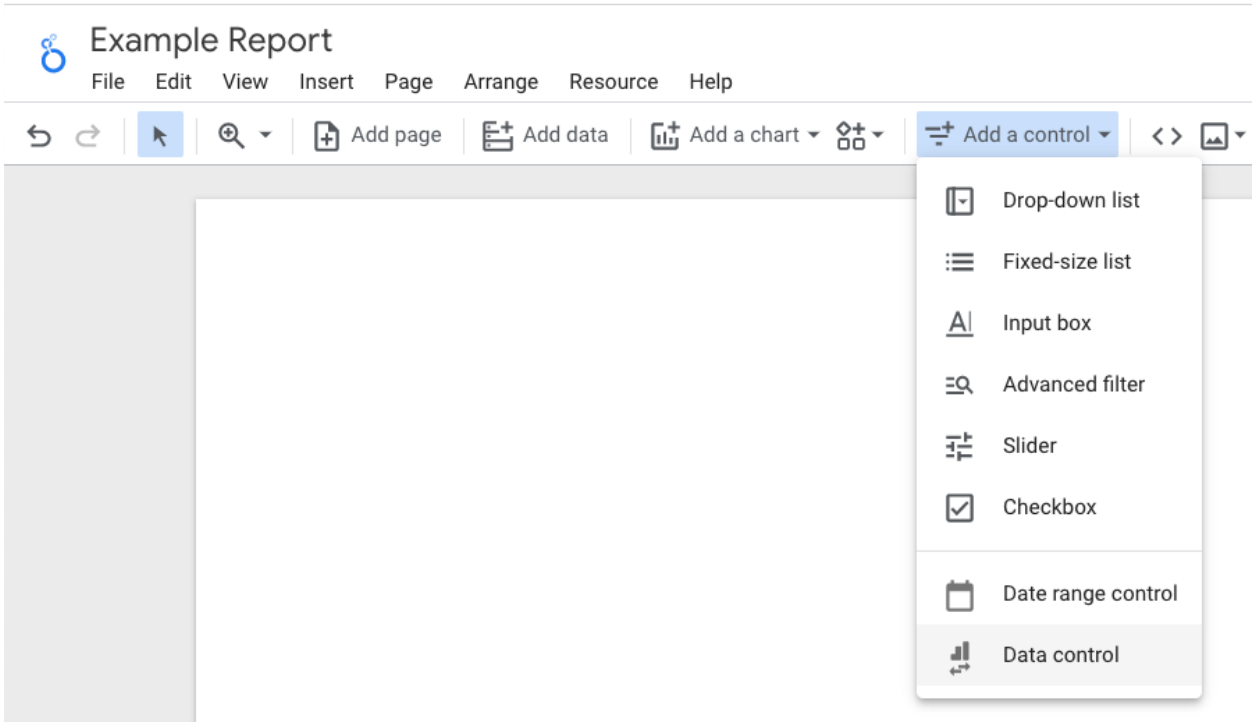
The next step is to build a report by clicking on “Blank Report”. Alternatively, templates can be used, however for this exercise, a blank report is selected:

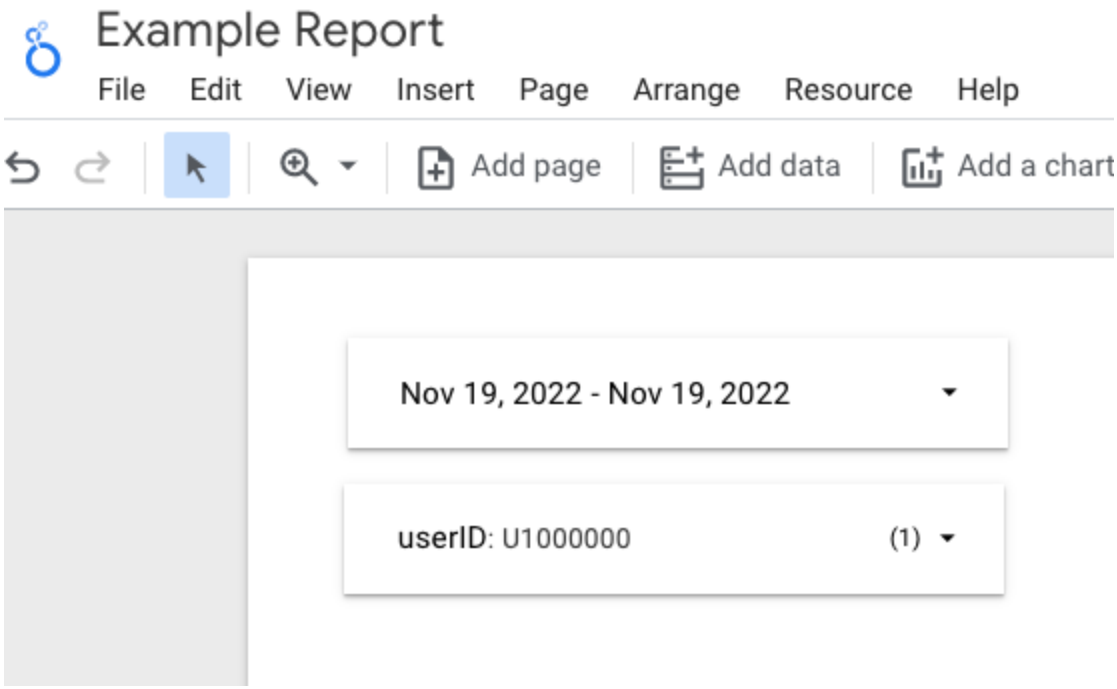


Once the report is initialised, Looker will prompt for a data source - and here, we can see the newly created data sources under “my data sources”:



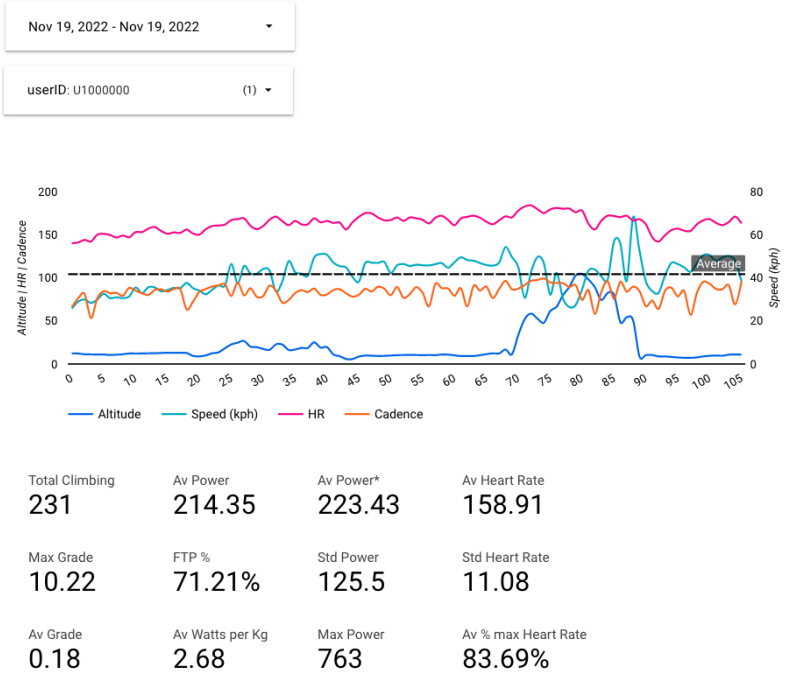
From here, a few data controls are used by dragging them into the report envelope. Date control and userID drop-down list are selected. Moreover, date range control properties are updated along with drop-down list properties are updated; the date is set as the 19th of November as the target date and userID is set as U1000000





From there, data charts are added to the report:

1. A time-series/line showing altitude and speed:
2. A few summary statistics



Total Climbing	Av Power	Av Power*	Av Heart Rate
231	214.35	223.43	158.91
Max Grade	**FTP %**	**Std Power**	**Std Heart Rate**
10.22	71.21%	125.5	11.08
Av Grade	**Av Watts per Kg**	**Max Power**	**Av % max Heart Rate**
0.18	2.68	763	83.69%

Research - User Interface Design

Two key business/platforms were used to inform User Interface Design i.e., what the report actually looks like, there are:

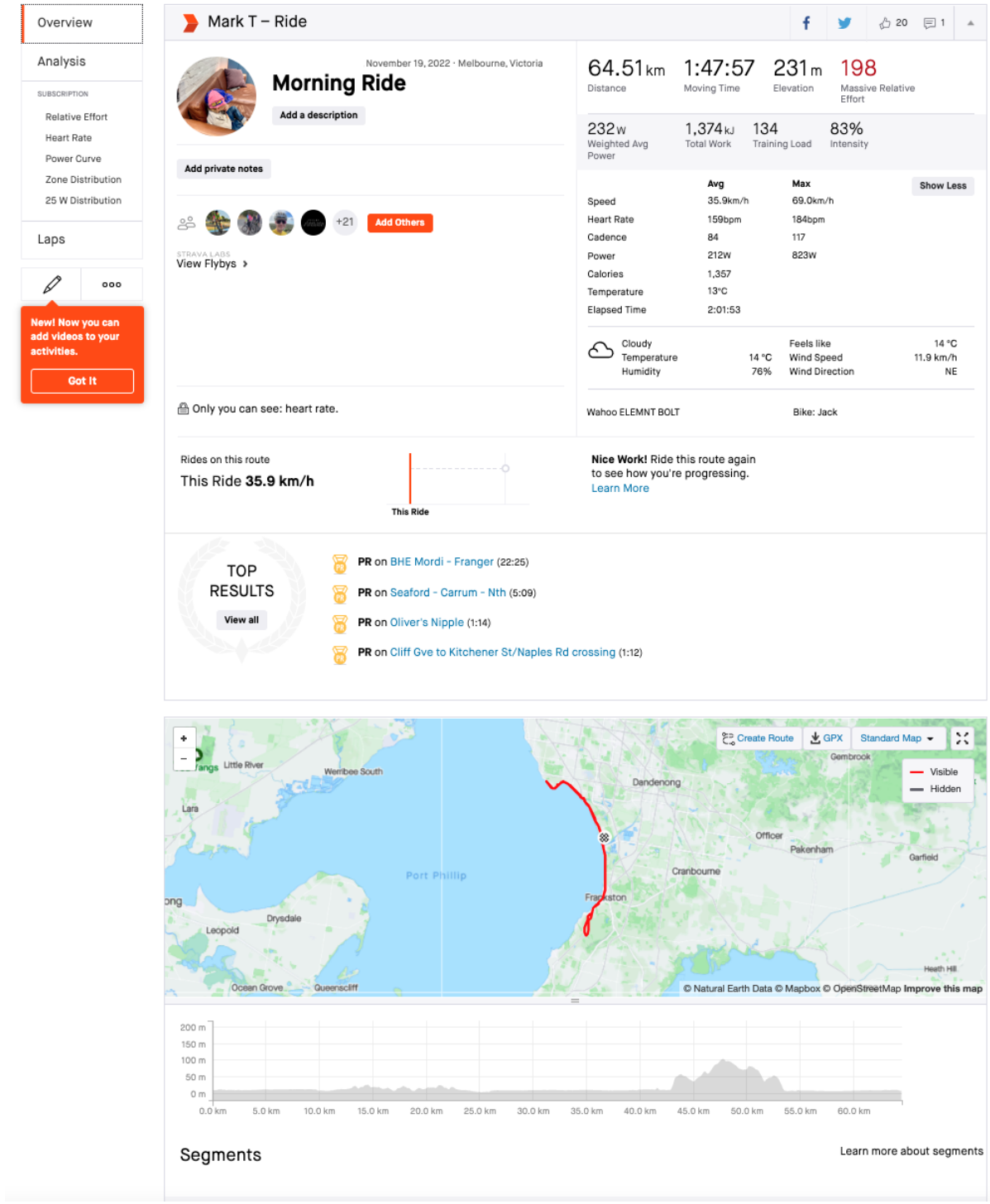
1. Strava
2. Wahoo

Strava

Strava is a widely used fitness social media app that records workout data along with a number of other data points. Moreover, Strava provides ‘Overview’ and ‘Analysis dashboards via their web app, see next page

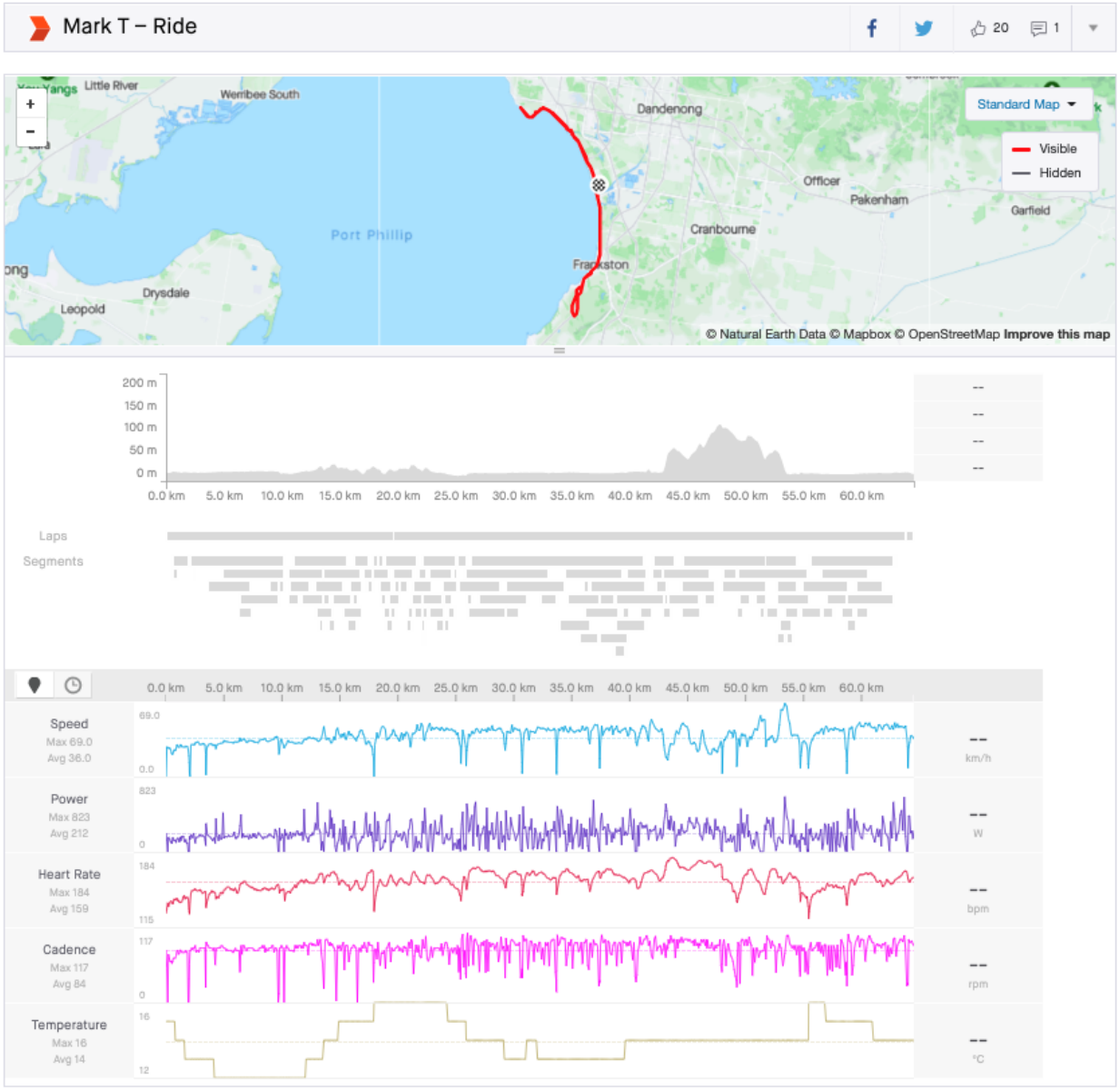
Overview Report

Provided by M.Telley



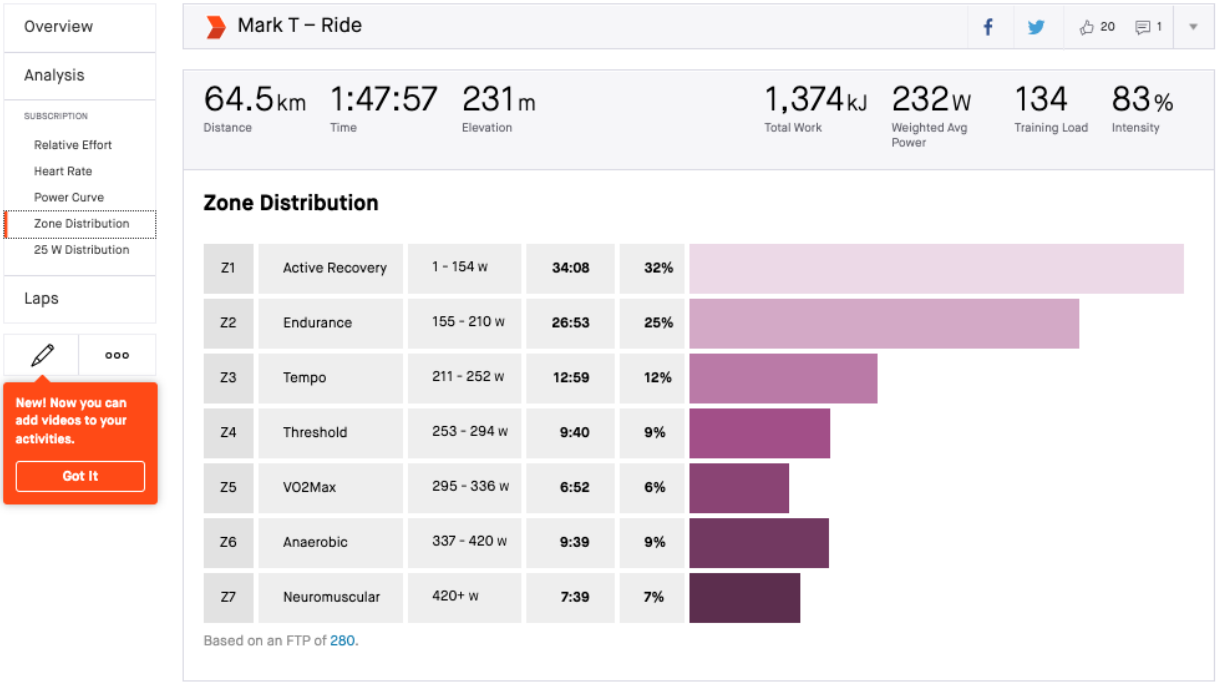
Analysis Report

Provided by M.Telley



Zone Distribution Report

Provided by M.Telley



Many of the features in the Strava dashboards were evaluated for implementation within the prototype dashboard. These were:

1. Zone distribution.
2. Time Series chart showing power, altitude, speed etc
3. Summary Statistics

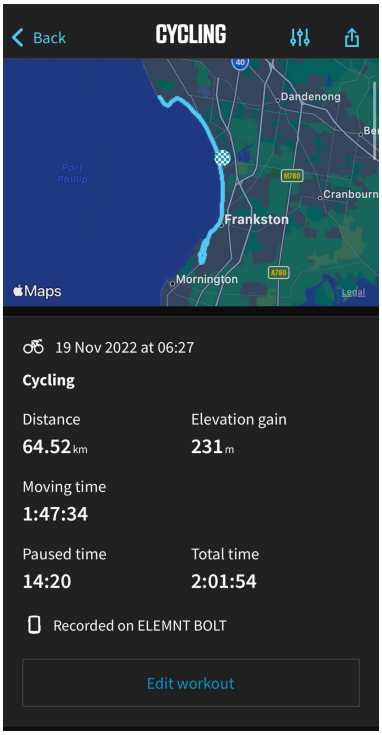
Wahoo

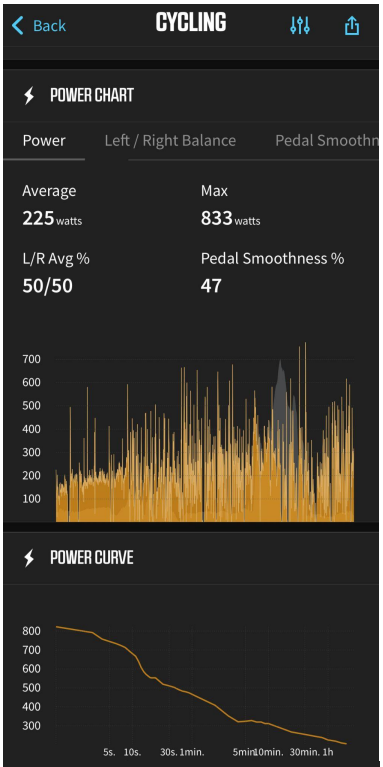
As listed on wahoo.com,
Founded *in 2009 by Chip Hawkins in Atlanta, GA, Wahoo creates innovative solutions to make hard-fought goals attainable and lives better. Wahoo was built on the foundation of simplicity and the mindset that "there's got to be a better way.'*

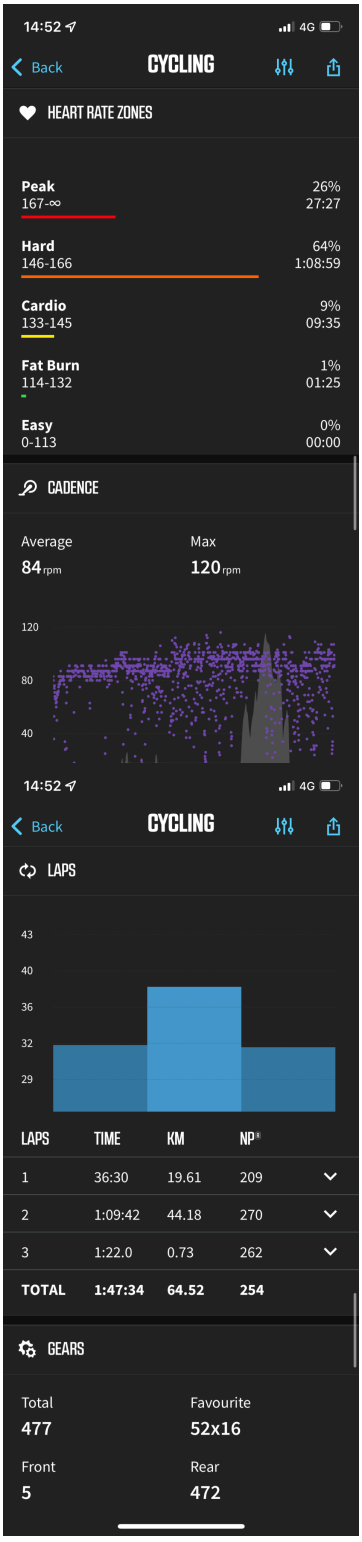
Wahoo manufactures a range of exercise-related hardware and software, some of which are used by Redback Operations i.e. Wahoo Kickr. Moreover, Wahoo has developed a world class software application called "Element" which is available both on iOS and Android devices that summarises the user's workout data.

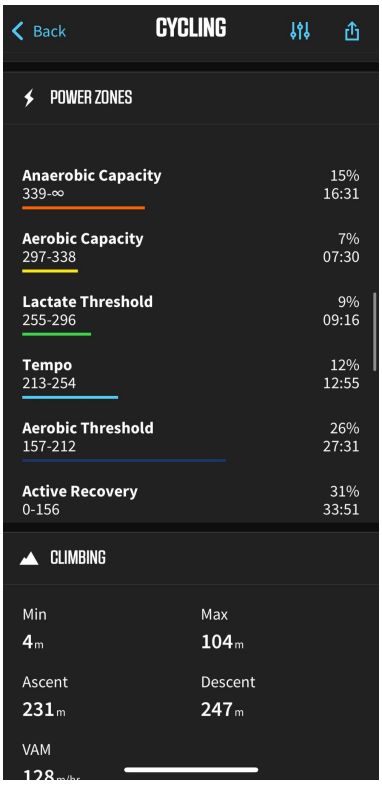
Interface Reports and Examples

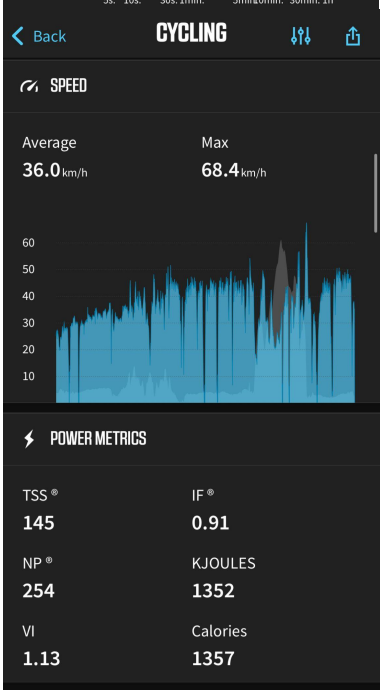
Provided by M.Telley











Similar to Strava, many of the features in the Wahoo dashboards were evaluated for implementation within the prototype dashboard. These were:

4. Zone distribution.
5. Time Series chart showing power, altitude, speed etc
6. Summary Statistics

Implementation and Design Process

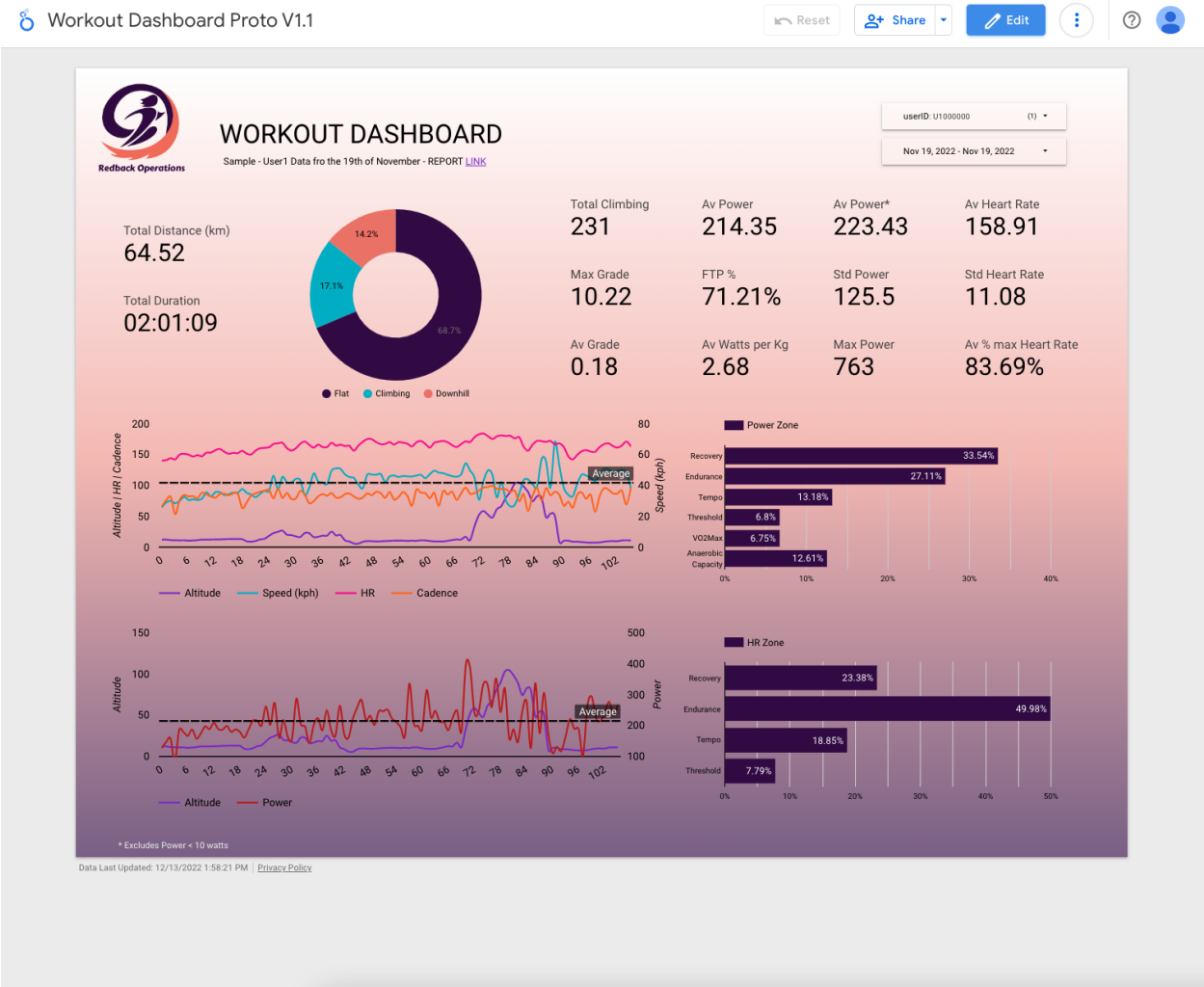
Based on the competitor research, a pathway to developing a minimal viable product was established - Three key data views or charts were targeted initially. Moreover, it was clear that certain features would require additional time to develop or wouldn't immediately be possible due to privacy concerns such as but not limited to

1. Map visibility (privacy concerns)
2. Total 'moving time'
3. Total 'pause time'
4. Total work (kj)
5. Weighted Avg Power
6. Weather data
7. Intensity
8. Training Load
9. Gearing details
10. Certain Climbing details
11. L/R Balance

Some of the features above could be explored in proceeding trimesters.

Final Product (MVP)

After development, along with the inclusion of additional data points such as “zone category”, and “heart rate category” as a means of establishing bins for the bar charts, the dashboard was complete



Conclusion

By undertaking thorough research of competitor products, and reviewing the various data points, chart types, data visualisations the MVP dashboard “Prototype 1” was completed. The final product, while still very basic, mimics reports available in the market and offers an example for other teams with Redback Operations about what is possible in relation to the availability of data points and how that translates into a user-friendly and concise dashboard concerning workout analysis. The report has a huge runway for further refinement and development over the proceeding trimesters.

Results

A basic but functional dashboard was built, tested and published.

Modeling

Redback Operations

Data Science and Analytics Team
M.TELLEY

DOCUMENTATION
ML Algorithm and Models

Purpose

Currently, Strava has a working model that will predict power output, however, on review it is quite inaccurate; Strava uses a mix of wind resistance, rolling resistance, gravity and acceleration. This is the frustration of many Amateurs cyclists using the application. As part of the Workout Analysis, prediction models could be used where certain data points were not immediately available i.e, power. As an example, power meters on certain indoor trainers and cycles are not factory fitted and are typically cost-prohibitive to purchase. Moreover, power or wattage output is a critical performance metric in performance cycling and can heavily influence a cyclist's training behaviour.

Scope

Investigate at minimum two different algorithms/models that could be used to predict power output. Based on research the following algorithms/models were selected:

1. Scikit-learn Linear Model - LinearRegression
2. Scikit-learn Linear Model - ElasticNet
3. Scikit-learn Random Forest - RandomForestRegressor

Prior to working on the implementation of the said algorithms/models listed above, vigorous data analysis needed to be completed to begin understanding and formulating a set of meaningful features and become familiar with key data trends, dependencies and relationships.

Data Analysis

A total of four different python projects using PyCharmEdu were created. As each project was completed, the level of data analysis increased - This was to ensure a steady learning pace both with the data and using Python and key libraries such as but not limited to Matlab, Pandas, Numpy etc. Moreover, an attempt was made to remotely query via API the BigQuery database to simulate a real-world environment i.e, assuming the Database is being consistently updated with new data, static local CSV files will not suffice.

1 - Big Query API and rudimentary data analysis

File: 1_bigquery_stat_analysis

The first seconds of the project aim to establish a connection with the BigQuery database, query the database for User 1 data and then store the information in a pandas dataframe. Note, the tables in the BigQuery had already undergone preprocessing in order to simplify the query requests in PyCharm.

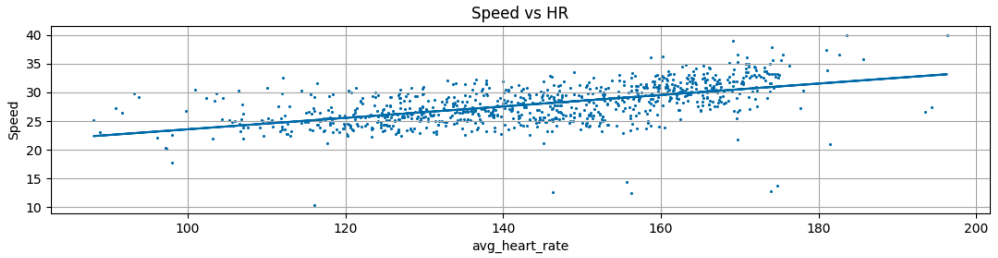
```
#Get Data
sql_query = """
    SELECT
        *
    FROM

`BIGQUERYTABLE.fitness_user_summary_features.feature-summary-filtered
`
    WHERE avg_power IS NOT NULL AND enhanced_avg_speed IS NOT NULL
 """

#Get User 1 Data
sql_query = """
    SELECT
        *
    FROM

`BIGQUERYTABLE.fitness_user_summary_features.feature-summary-filtered
-user1`
    WHERE avg_power IS NOT NULL AND enhanced_avg_speed IS NOT NULL
"""
```

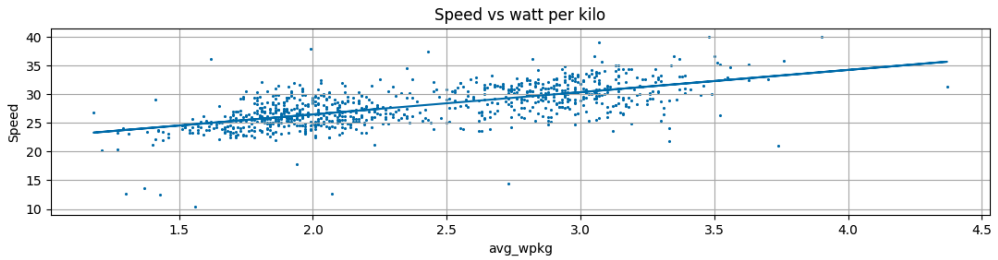
Once data for both all available users and User1 separately were stored in respective dataframes, correlation of numeric data attributes was conducted using Pandas Corr() function, to begin to understand the relationships (if any) between the data attributes.

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html

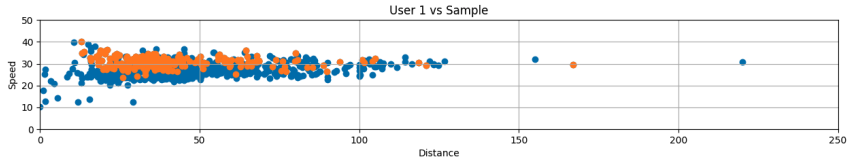
```
#Reformat DF to have only numeric data attributes
dfCor = df2[['distance',
             'enhanced_avg_speed',# 'avg_power',
             'avg_wpkg',
             'avg_power',
             'avg_heart_rate',
             'avg_max_heart_rate_perct',
             'day_of_week',
             'hour']]
matrix = dfCor.corr()
print(matrix)
corrplot(matrix, size_scale=500, marker='s')
matrix.to_csv("data_corr.csv")
```

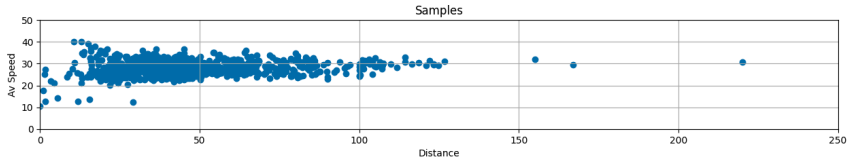
Once the correlation analysis was complete, attempts at visualising relationships of interest were conducted for - Speed vs Heart Rate and Speed vs Watts per Kilogram

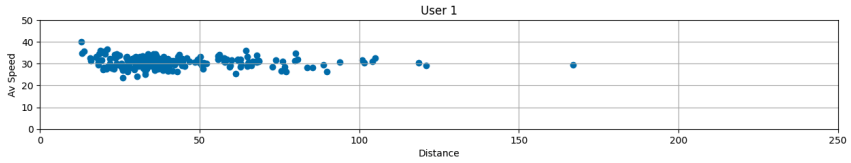




After plots were created and shown, an attempt to understand how the target user's (User1) data metrics differed (if any) from the sample data metrics:







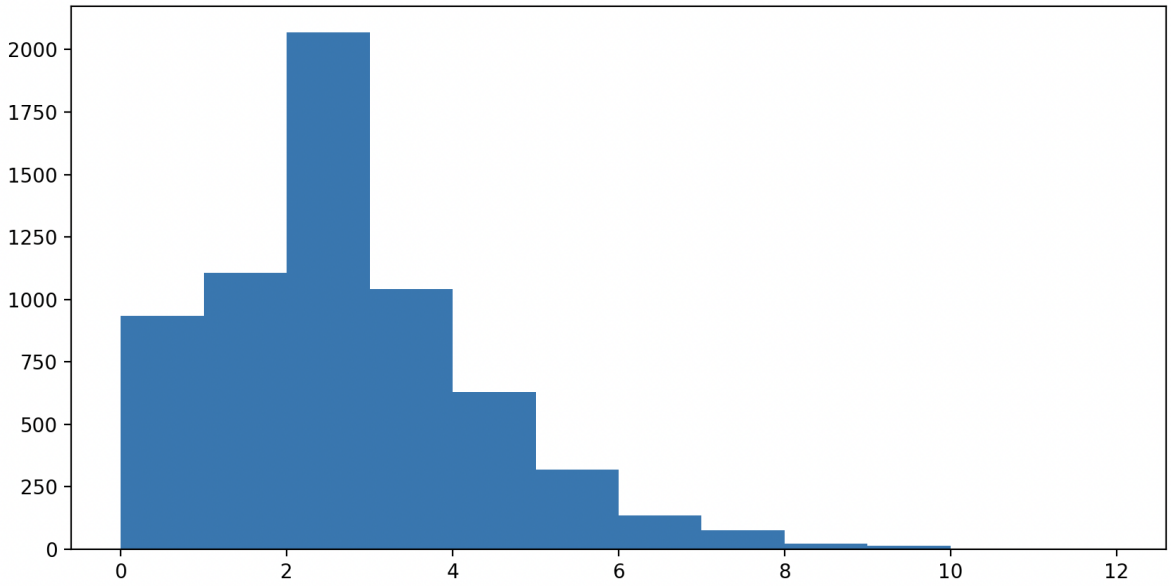
This concluded the first attempt at a) connecting via API to the Bigquery database to query, pull and store data and then a few minor attempts at analysing the data available.

2 - Supplementary Bigquery and Data Analysis

File: 2_workout_data_modelling

For the second attempt at using PyCharm to query the BigQuery database and conduct analysis work, a much larger dataset was used; the number of rows = >6,000.

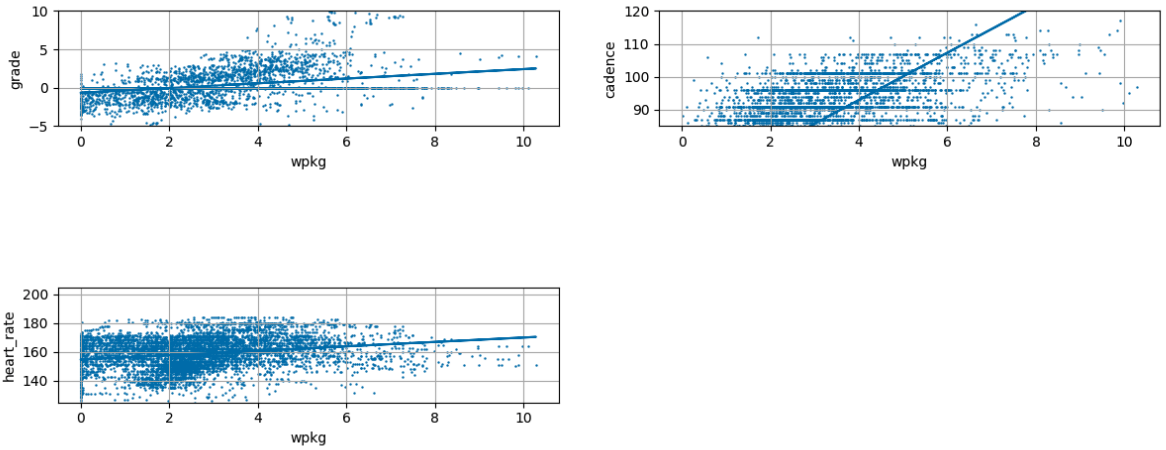
The initial focus was to understand the distribution of power using a standardised measure called Wattage per kilogram (wpkg) - The plot below showed the distribution of wpkg was predominantly between 2 and 3 wpkg - understand what a ‘normal’ wpkg value was going to be important in the future.



Thereafter, a correlation plot was shown to aid in visualising the analysis of key performance metrics. The following resource was used: https://pypi.org/project/heatmapz/. It was clear that cadence was becoming a data metric of key interest in relation to power-related metrics (wpkg,

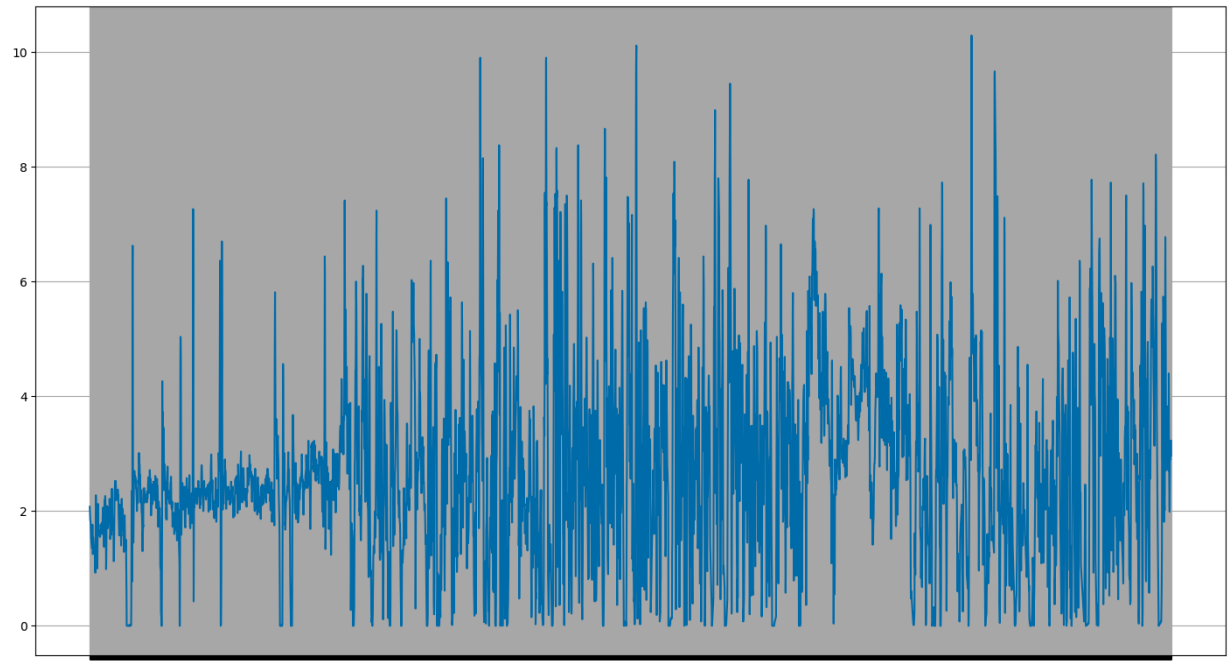
power and the % of functional threshold power - FTP).

The top three metrics of interest were then plotted in a scatter plot against wpkg to again, aiding visualise data analysis:



To finish off this PyCharm project, an attempt was made to plot over power data - This presented problems as there were too many records to present and the deviation between the

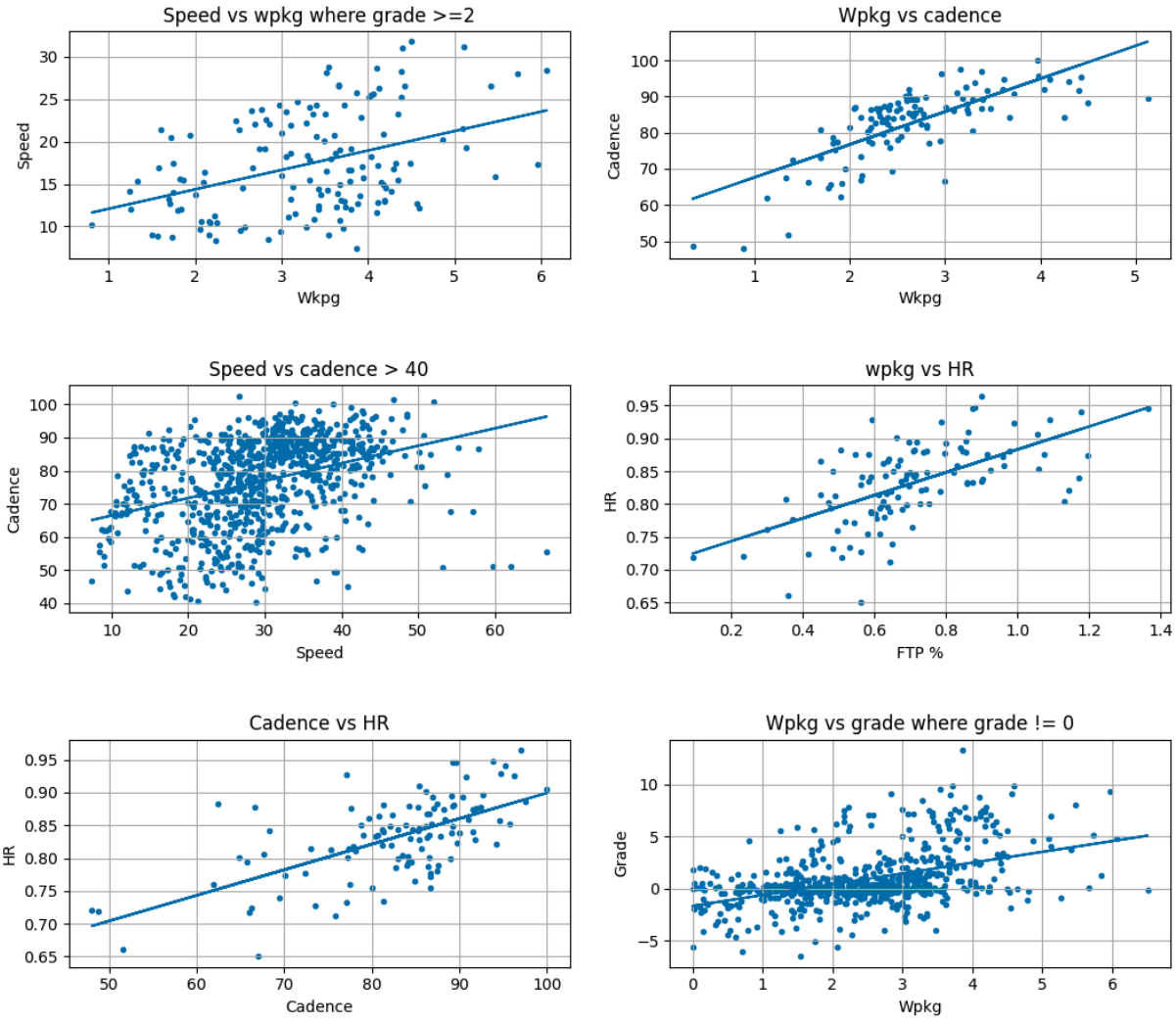
data points produced a very noisy plot. The data was recorded in intervals of 1 second - Further work was needed to get data into different time intervals.



3 - Continuation of the Previous Project

File: 3_bigquery_summary_stats

Further work was conducted in BigQuery to wrangle data into a format that would be conducive to future data analysis. The big change was getting activity data into 1-minute intervals along with being to focus on a single activity session (had defined start and end points). To begin with, correlations were assessed again to determine if anything had changed and a larger number of performance metrics were assessed:



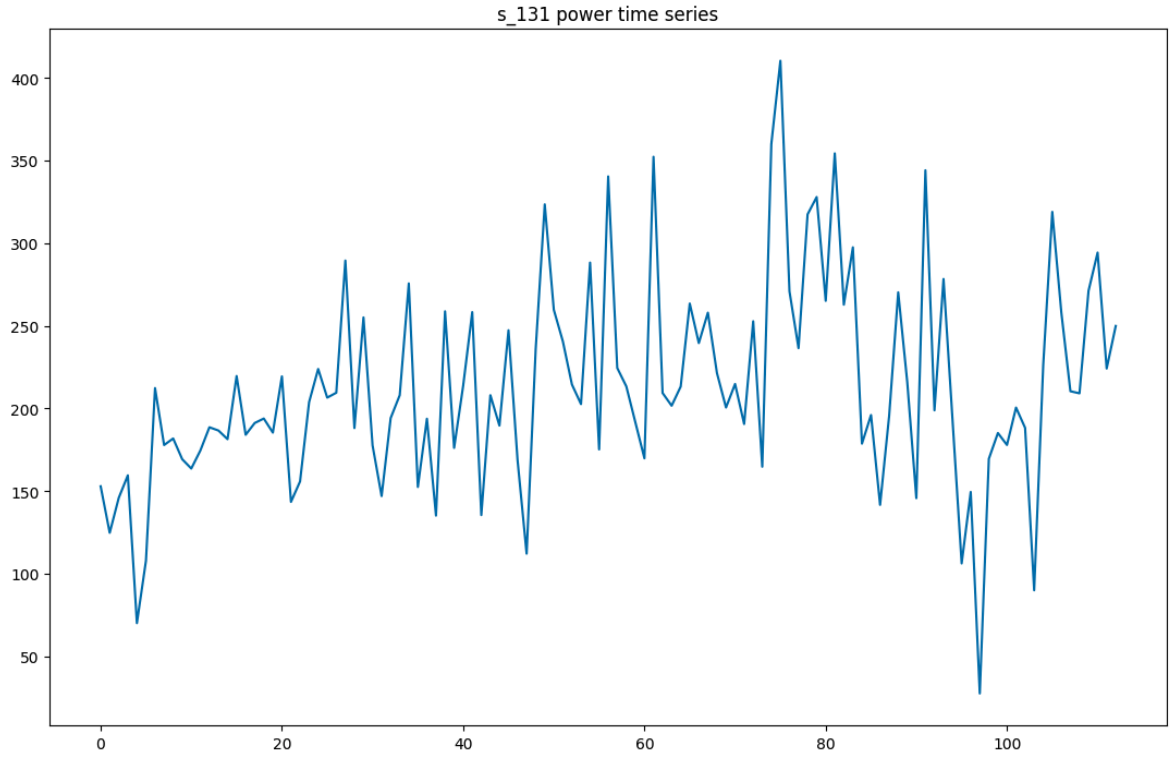
Unlike previous attempts at Data Analysis, filtering of data did occur:

1. grade != 0 - An attempt was made to assess how grade influenced power where the grade was not zero. The bottom right plot above shows a somewhat strong linear relationship ie when a cyclist is riding up an incline it is likely that power output will be higher

2. Grade > 2 - Attempt to assess the impact grade has on speed and wpkg where the grade is > 2; it was clear that a higher speed was positively associated with high wpkg.
3. Cadence > 40 - In an attempt to remove idling movement i.e., when cadence is equal to zero or very low (<40), the relationship between speed and cadence was assessed. There was a weak positive relationship between the two variables.

The outcome of future analysis helps establish a stronger view of the key elements that might influence power output.

To finish, a plot of power output (watts) for a single session (**session # 131**) was attempted - Session 131 will be used later as the key dataset to compare model performance. However, unlike previous attempts, the time interval or step was set at 1 minute. The plot below is clearer and more readable and forms the basis for future analysis. The ability to potentially test and learn in 1-second intervals but display in 1-minute intervals would later be established.



This concluded data analysis efforts.

4/ Modeling

Scikit-learn Linear Model - LinearRegression

File: Elastic-Net-Regression-Models-Workout-Data

To begin and as identified earlier, a basic Linear Regression model would be adopted initially. The first parts of this project focus on statistical analysis building on previous data analysis efforts. Moreover, the dataset used for this model was initially adopted in 5-minute intervals to aggregate (average metric values over a 5-minute period) the data to reduce the deviation in data attribute values and the power metrics selected were watts as opposed to watts per kilogram. Thereafter the standard 1-second interval would be used.

Linear regression is the most basic form, where the model is not penalised for its choice of weights, at all. That means, during the training stage, if the model feels like one particular feature is particularly important, the model may place a large weight on the feature or data attribute. This may become problematic as the size of the data set becomes larger (Wenwei Xu)

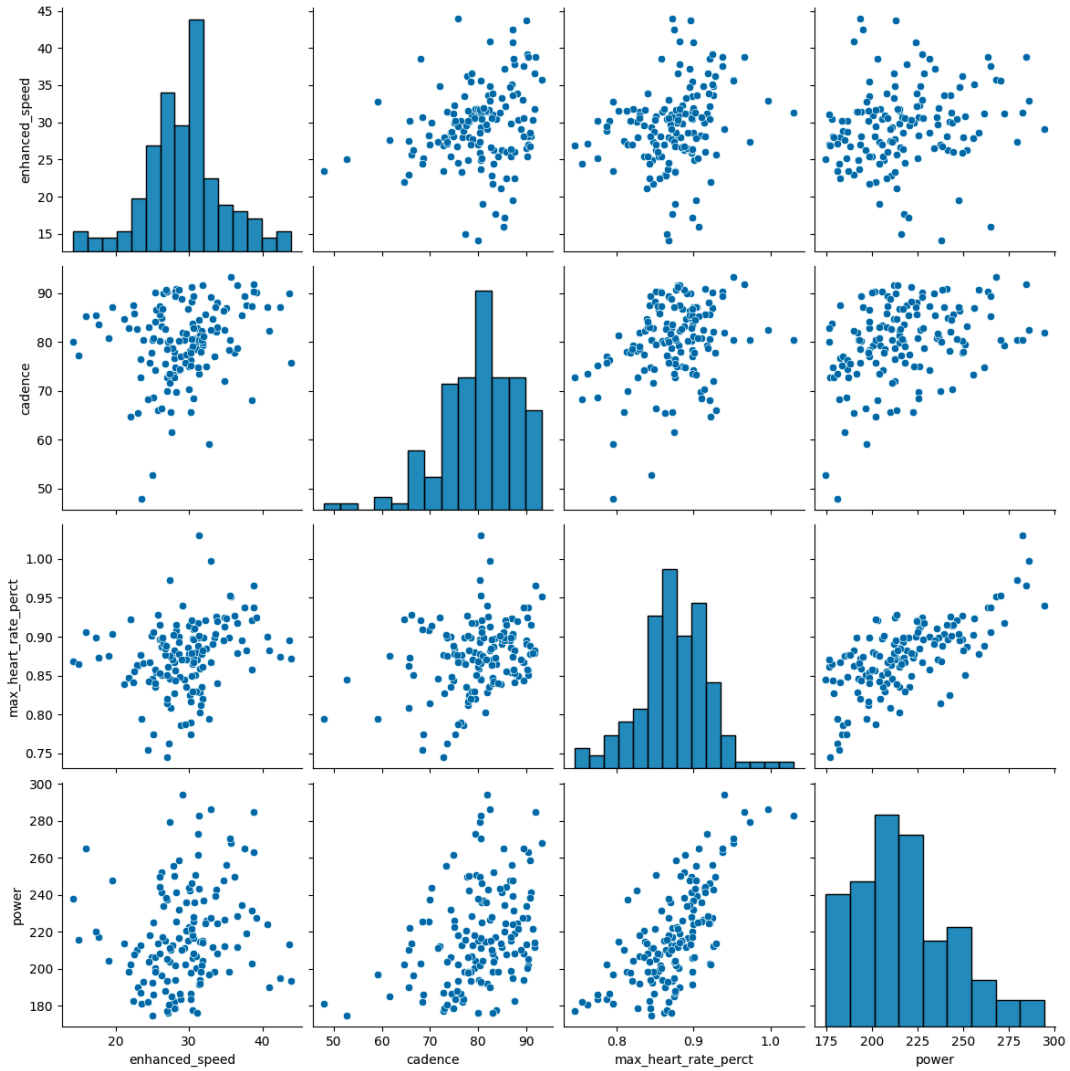
Dataset dynamics:

- Train set (5-minute): Over 12 hours of activity were used, equating to over 145 rows of data.
- Test set (5-minute): Only 48 hours of activity were used, equating to over 580 rows of data.
- Train set (1-second): Over 9 hours of activity were used, equating to over 33,750 rows of data.
- Test set (1-second): Over 37.5 hours of activity were used, equating to over 135,000 rows of
- data.

The increase in the amount of data was also a considered effort - The impact to model performance where data was excessively more abundant would be monitored.

5 minutes data set

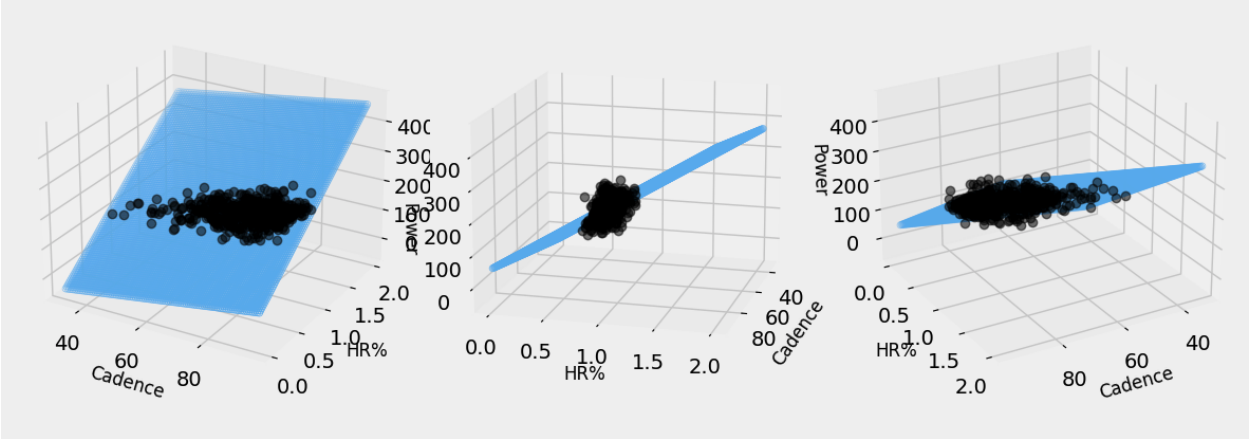
A quick assessment of metric correlations and distribution was calculated:



Over the available data attributes - cadence and % of max heart rate (max heart rate is calculated using the HUNT formula, Age = 33). These two attributes would be used in the model.

Outputs

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  power   R-squared:                       0.453
Model:                            OLS   Adj. R-squared:                  0.451
Method:                 Least Squares   F-statistic:                     239.3
Date:                Tue, 13 Dec 2022   Prob (F-statistic):           2.10e-76
Time:                        20:23:10   Log-Likelihood:                -2709.4
No. Observations:                 580   AIC:                             5425.
Df Residuals:                     577   BIC:                             5438.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        -90.6007     16.413     -5.520      0.000    -122.838     -58.364
cadence        1.5224      0.114     13.367      0.000       1.299       1.746
max_heart_rate_perct   204.2133     20.591      9.918      0.000     163.772     244.655
==============================================================================
Omnibus:                        8.998   Durbin-Watson:                   1.127
Prob(Omnibus):                  0.011   Jarque-Bera (JB):               13.197
Skew:                           0.100   Prob(JB):                      0.00136
Kurtosis:                       3.711   Cond. No.                     1.85e+03
==============================================================================
```



R^2 range between 0 and 1, where R^2=0 means there is no linear relationship between the variables and R^2=1 shows a perfect linear relationship. In our case, we got an R^2 score of 0.45 which means 45% of our dependent variable can be explained using our independent variables.

Model Validation

We can evaluate a model by looking at its coefficient of determination (R^2), F-test, t-test, and also residuals. Before we continue we will rebuild our model using the statsmodel library with

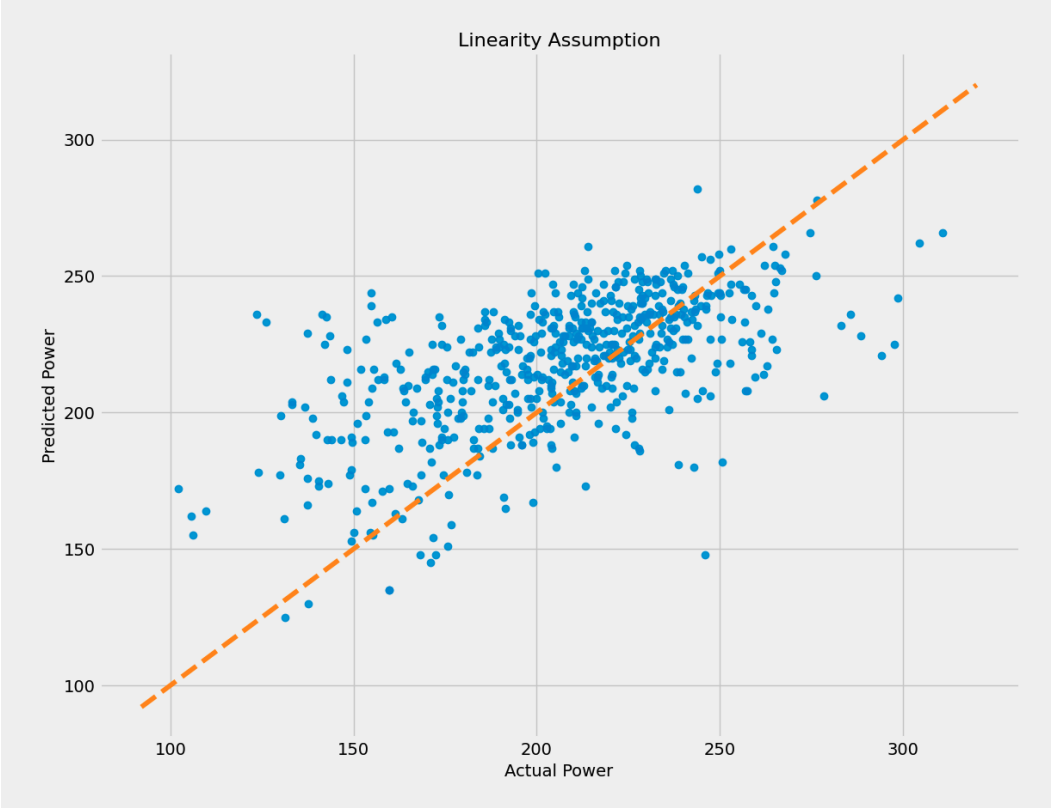
the OLS() function. Then we will print the model summary using the summary() function on the model. The model summary contains lots of important values we can use to evaluate our model.

Prediction = Intercept - Coefficients1-x_1 + Coefficients2-x_2. The intercept value is the estimated average value of our dependent variable when all of the values of our independent variables are 0

Intercept: -153.09521518
Coefficients: 0.50632036, 378.11543937
Model R^2: 0.4533
F-statistic: 239.28687410590658
Probability of observing value at least as high as F-statistic: 2.10

Linearity

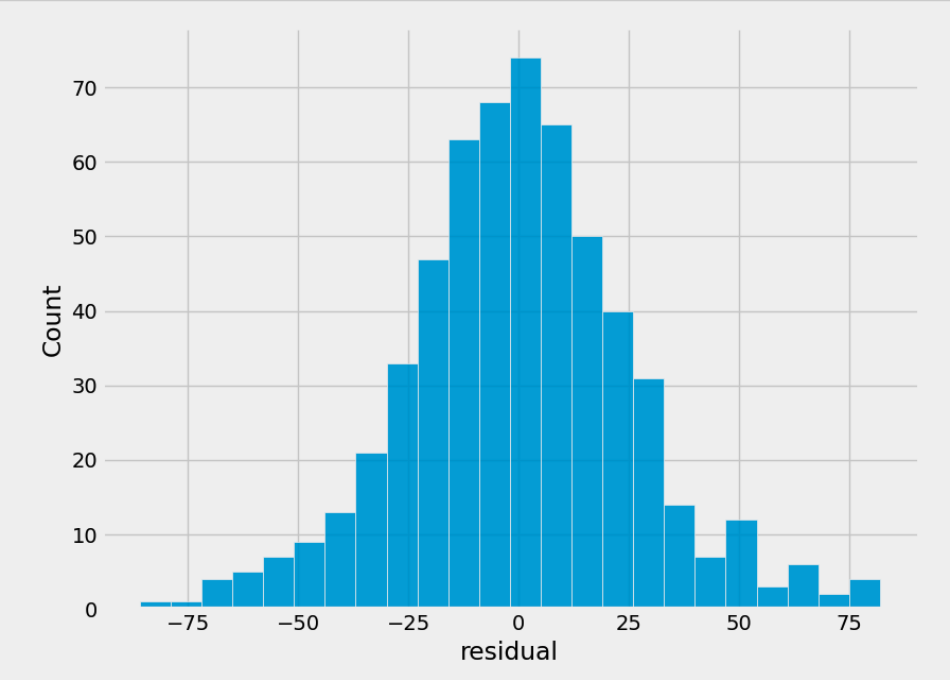
This assumes that there is a linear relationship between the independent variables and the dependent variable. In our case since we have multiple independent variables, we can do this by using a scatter plot to see our predicted values versus the actual values. Plotting the observed vs predicted values



Normality

This assumes that the error terms of the model are normally distributed. We will examine the normality of the residuals by plotting it into a histogram and looking at the p-value from the Anderson-Darling test for normality. We will use the normal_ad() function from statsmodel to calculate our p-value and then compare it to the threshold of 0.05, if the p-value we get is higher than the threshold then we can assume that our residual is normally distributed.

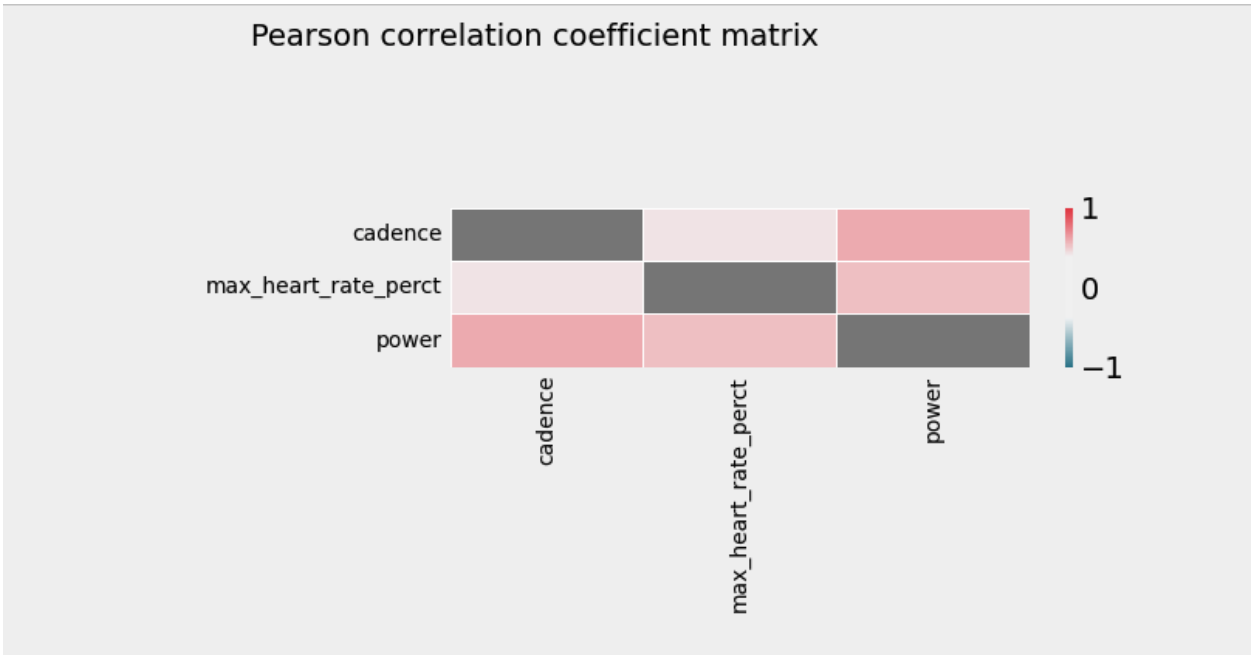
p-value from the test Anderson-Darling test below 0.05 generally means non-normal: 0.00047. Residuals are not normally distributed



Multicollinearity

This assumes that the predictors used in the regression are not correlated with each other. To identify if there are any correlations between our predictors we can calculate the Pearson correlation coefficient between each column in our data using the corr() function from Pandas dataframe. Then we can display it as a heatmap using heatmap() function from Seaborn.

The output of the heatmap shows us that the independent variables are affecting each other and that there is multicollinearity in our data.



Autocorrelation

Autocorrelation is the correlation of the errors (residuals) over time. Used when data are collected over time to detect if autocorrelation is present. Autocorrelation exists if residuals in one time period are related to residuals in another period. We can detect autocorrelation by performing the Durbin-Watson test to determine if either a positive or negative correlation is present. In this step, we will use the durbin_watson () function from statsmodel to calculate our Durbin-Watson score and then assess the value with the following condition:

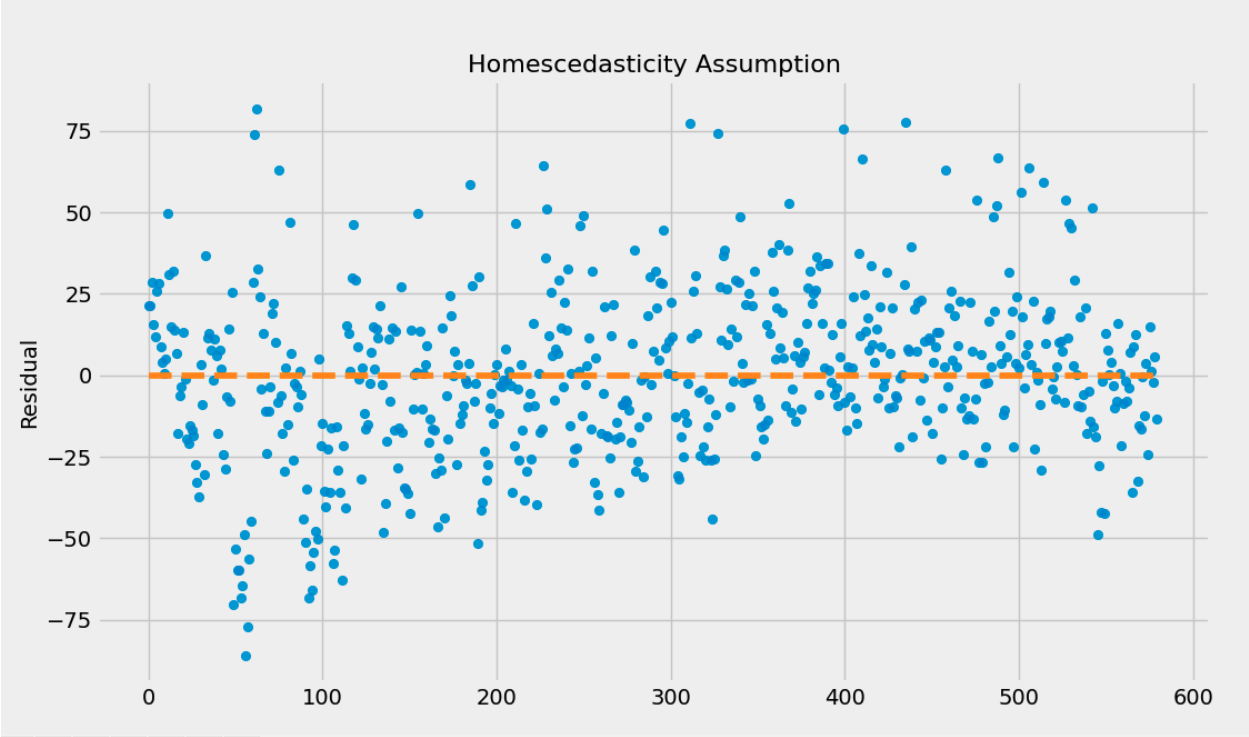
Outcomes

1. If the Durbin-Watson score is less than 1.5 then there is a positive autocorrelation and the assumption is not satisfied
2. If the Durbin-Watson score is between 1.5 and 2.5 then there is no autocorrelation and the assumption is satisfied
3. If the Durbin-Watson score is more than 2.5 then there is a negative autocorrelation and the assumption is not satisfied
4. We can assume that there are Signs of positive autocorrelation in our residual.

Durbin-Watson: 1.127 = Signs of positive autocorrelation; assumption not satisfied

Homoscedasticity

This assumes homoscedasticity, which is the same variance within our error terms. Heteroscedasticity, the violation of homoscedasticity, occurs when we don't have an even variance across the error terms. To detect homoscedasticity, we can plot our residual and see if the variance appears to be uniform - which they don't seem to be.

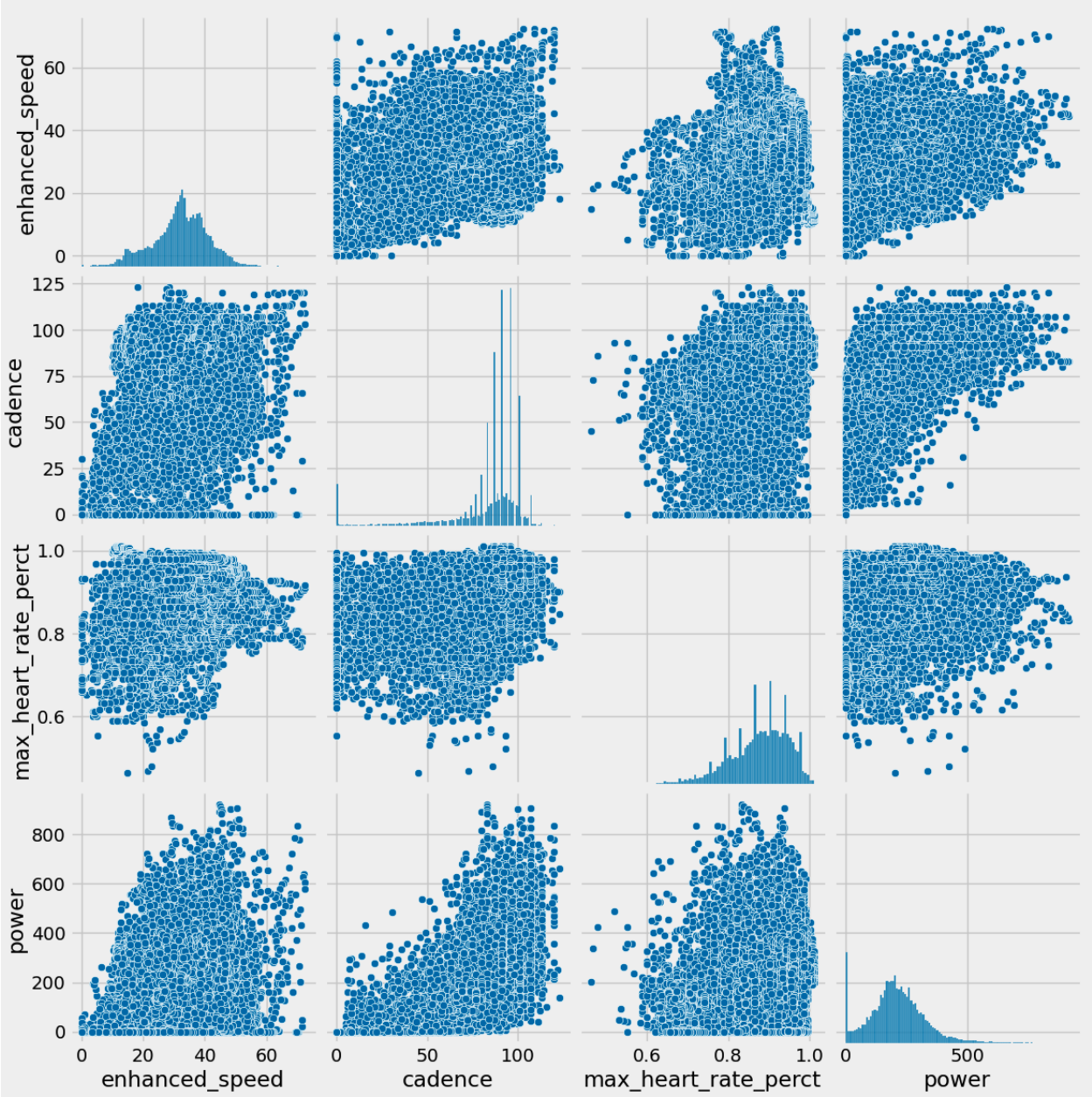


Results

Our model has failed to pass all the tests in the model validation steps, so we can't conclude that our model can perform well to predict power output using the two independent variables, cadence and max HR %. Our model only has an R^2 score of ~45%, which means that there are still about 57% unknown factors that are affecting our power output. This analysis will inform future efforts to predict power output

1 Second Dataset

A quick assessment of metric correlations and distribution was calculated:

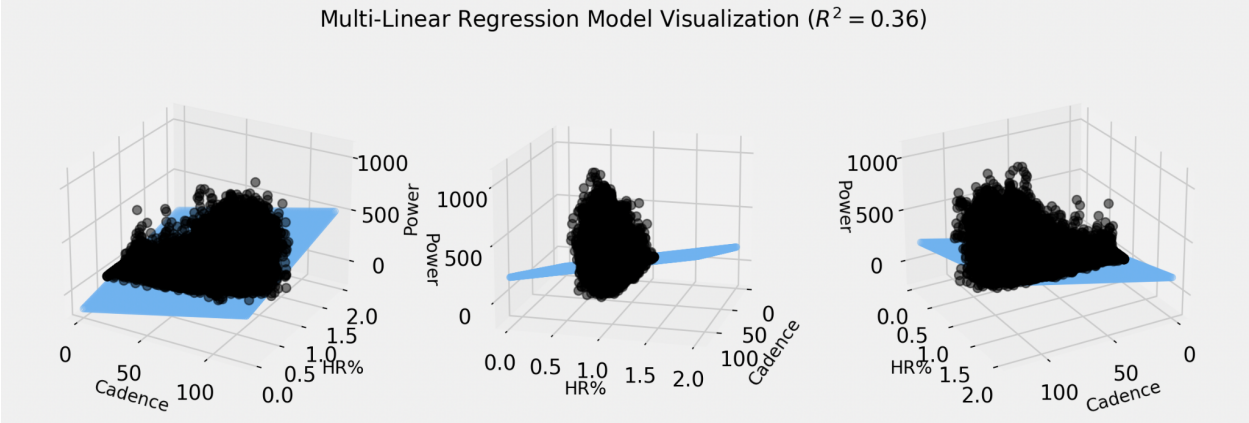


OLS Regression Results

Dep. Variable:	power	R-squared:	0.362
Model:	OLS	Adj. R-squared:	0.362
Method:	Least Squares	F-statistic:	3.846e+04
Date:	Tue, 13 Dec 2022	Prob (F-statistic):	0.00
Time:	20:54:27	Log-Likelihood:	-8.0161e+05
No. Observations:	135455	AIC:	1.603e+06
Df Residuals:	135452	BIC:	1.603e+06
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-152.2997	2.723	-55.929	0.000	-157.637	-146.962
cadence	2.8752	0.011	250.475	0.000	2.853	2.898
max_heart_rate_perct	151.4784	3.285	46.117	0.000	145.041	157.916

Omnibus:	49204.492	Durbin-Watson:	0.183
Prob(Omnibus):	0.000	Jarque-Bera (JB):	265726.393
Skew:	1.669	Prob(JB):	0.00
Kurtosis:	8.994	Cond. No.	1.43e+03



R^2 range between 0 and 1, where $R^2=0$ means there is no linear relationship between the variables and $R^2=1$ shows a perfect linear relationship. In our case, we got an R^2 score of 0.36 which means 36% of our dependent variable can be explained using our independent variables.

Model Validation

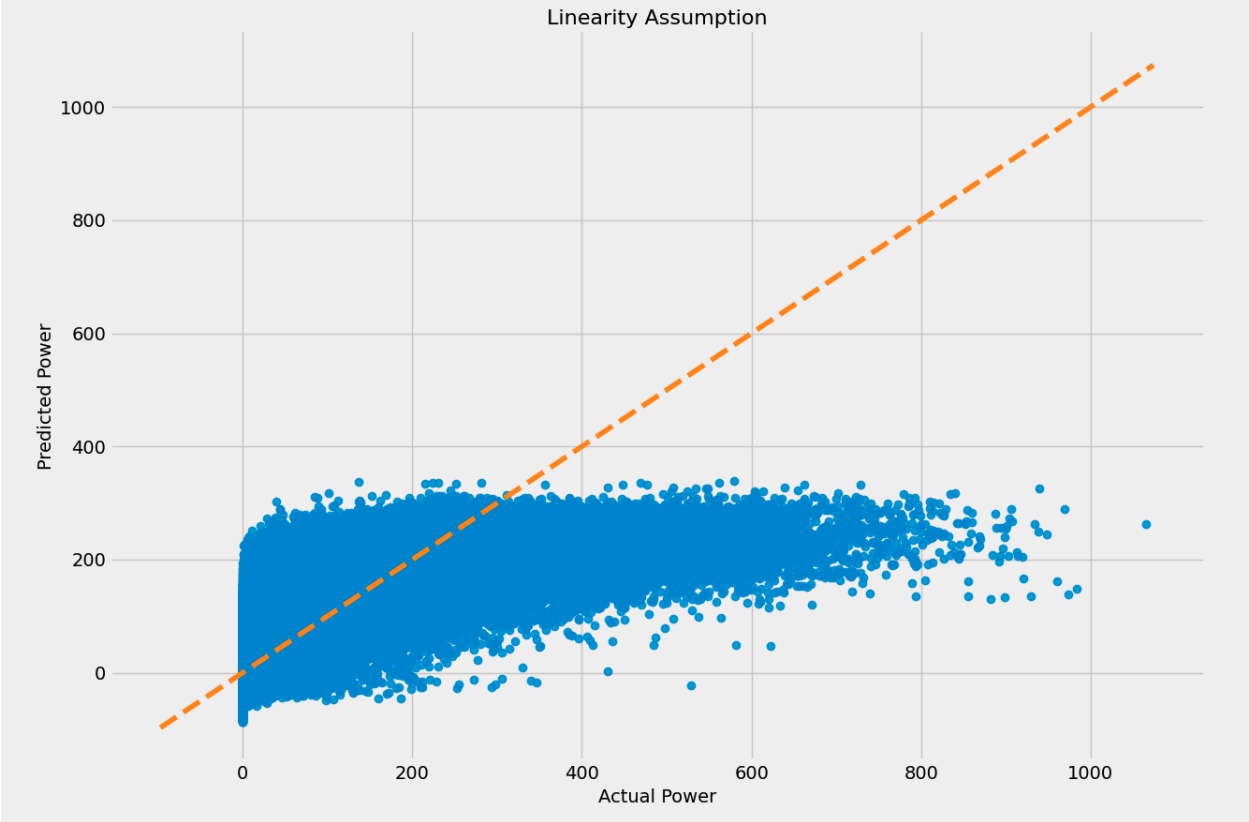
We can evaluate a model by looking at its coefficient of determination (R^2), F-test, t-test, and also residuals. Before we continue we will rebuild our model using the statsmodel library with the OLS() function. Then we will print the model summary using the summary() function on the model. The model summary contains lots of important values we can use to evaluate our model.

Prediction = Intercept - Coefficients1-x_1 + Coefficients2-x_2. The intercept value is the estimated average value of our dependent variable when all of the values of our independent variables are 0

Intercept: -131.63195875
Coefficients: 3.13720344 92.68999586
Model R^2: 0.362
F-statistic: 38455.55
Probability of observing value at least as high as F-statistic: 0.0

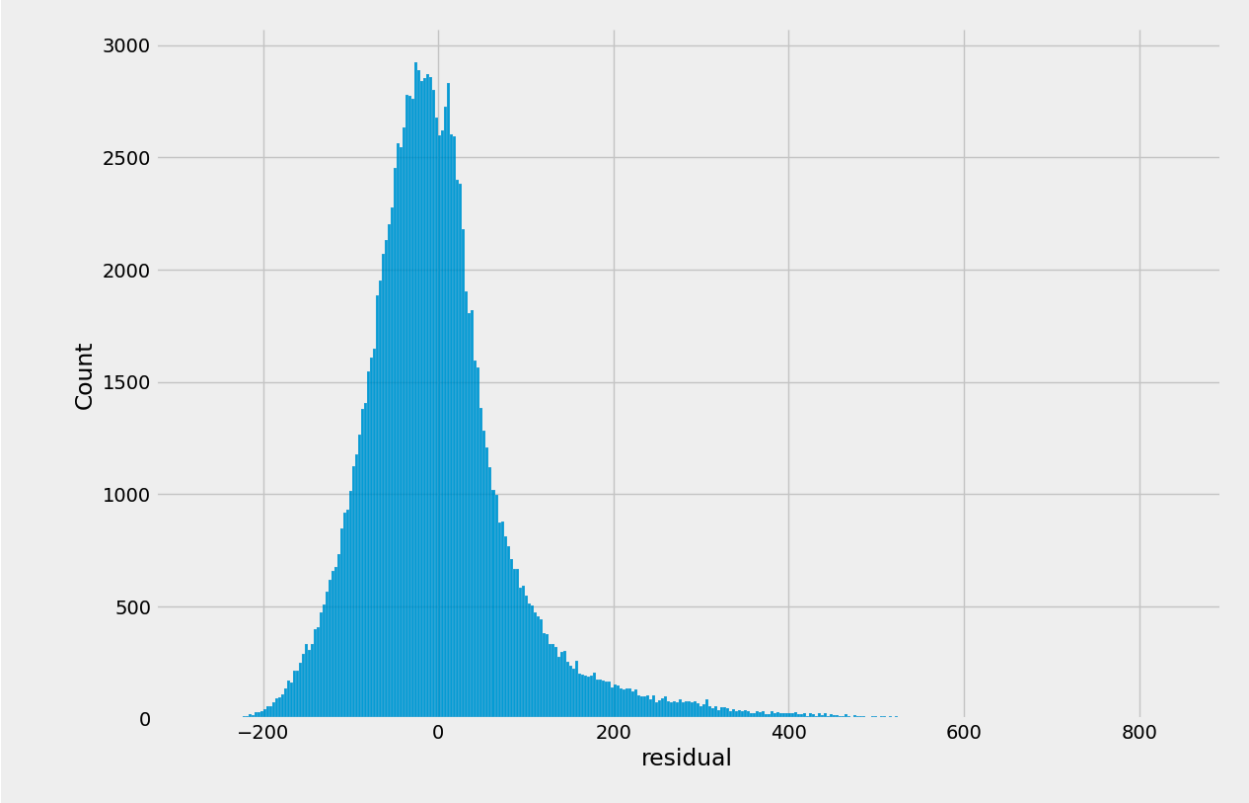
Linearity

The 1-second data set performs extremely poorly



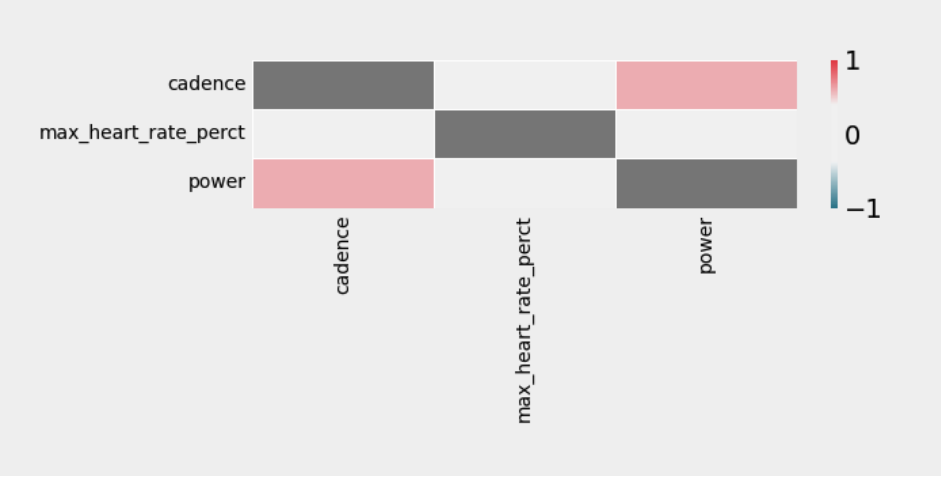
Normality

p-value from the test Anderson-Darling test below 0.05 generally means non-normal: 0.00. Residuals are not normally distributed



Multicollinearity

Unlike the 5-minute data set, the output of the heatmap shows us that the independent variables are not affecting each other and that there is no multicollinearity in our data.

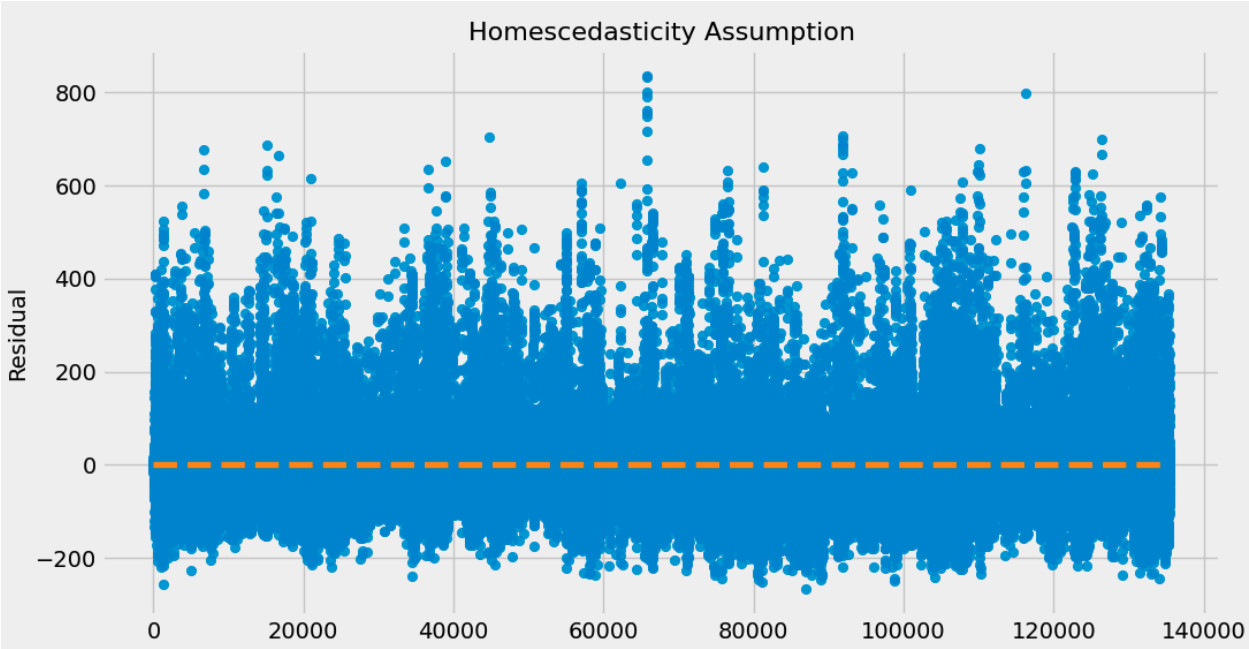


Autocorrelation

Durbin-Watson: 1.183 = Signs of positive autocorrelation; assumption not satisfied

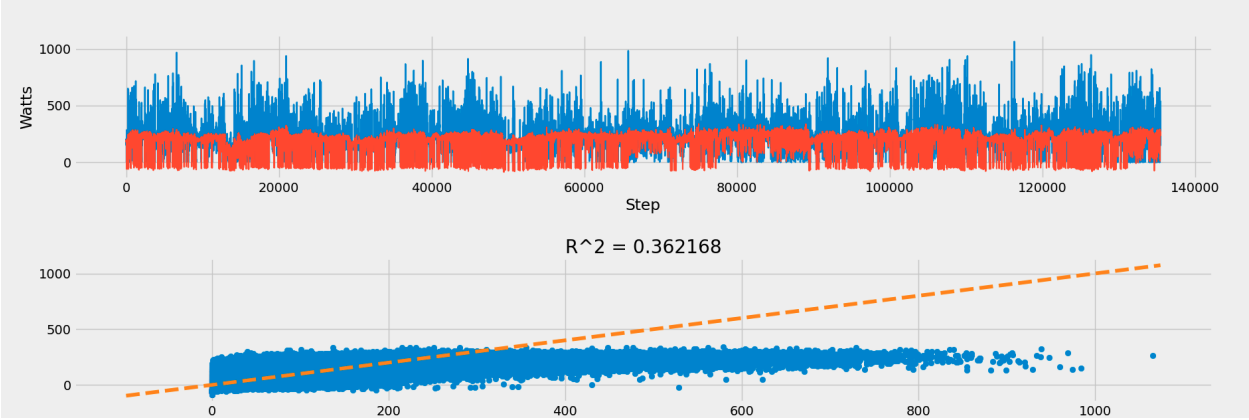
Homoscedasticity

To detect homoscedasticity, we can plot our residual and see if the variance appears to be uniform - which they don't seem to be.



Results

Our model has failed to pass all the tests in the model validation steps and has performed worse compared to the 5-minute data set, so we can't conclude that our model can perform well to predict power output using the two independent variables, cadence and max HR %. Our model only has an R^2 score of ~36%, which means that there are still about 64% unknown factors that are affecting our power output. This analysis will inform future efforts to predict power output:



Resource: https://medium.com/swlh/multi-linear-regression-using-python-44bd0d10082d

Summary

Linear Regression is not a good fit for this problem for a range of reasons; prediction needs to be highly accurate, the amount of data is too large (overfitting is a real problem), lack of penalty or rewards functions and there are a number data attributes (features) that are not being included in the model.

Scikit-learn Linear Model - ElasticNet

Elasticnet regression combines Ridge and Lasso regression into a single algorithm. The hyperparameters Alpha and L1 ratio, control the penalties imposed on the algorithm. For example, if L1 ratio =0 we have ridge regression, if L1 ratio = 1 we have lasso. ElasticNet should eliminate overfitting and improve overall accuracy.

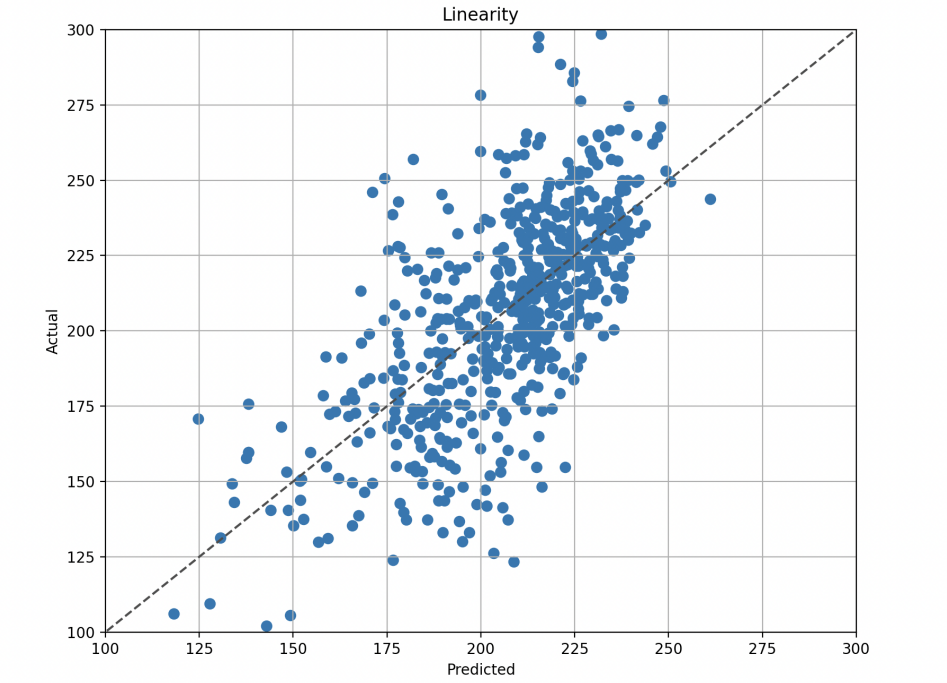
The same datasets are used across Elasticnet and LinearRegression.

Run before tuning Elastic Net Hyperparameters

5-minute data set

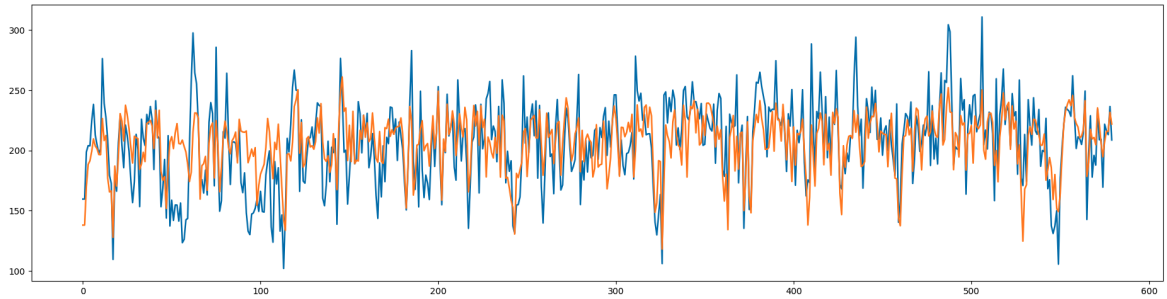
Linearity

Groupings are much tighter than before.



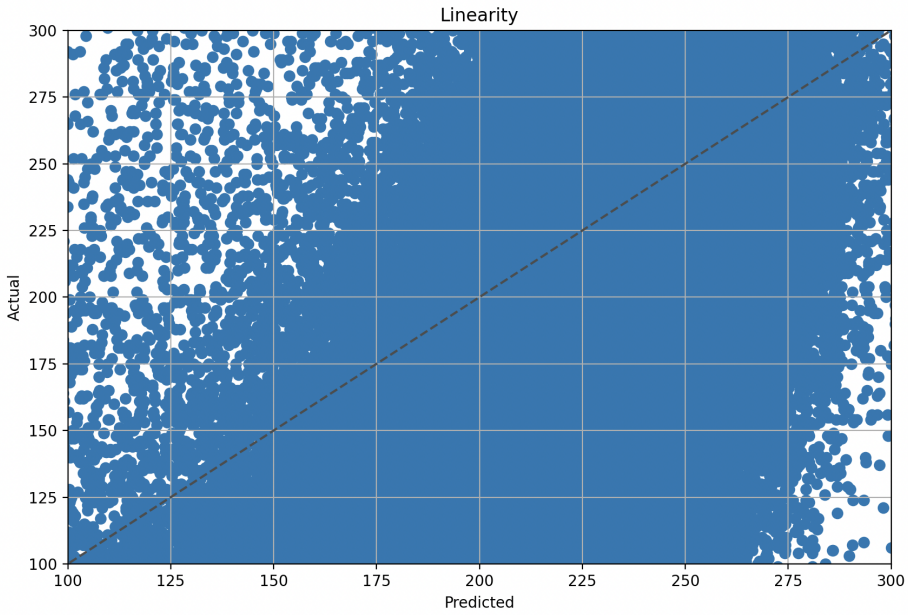
Predictions / Performance

$R^2 = 0.454$



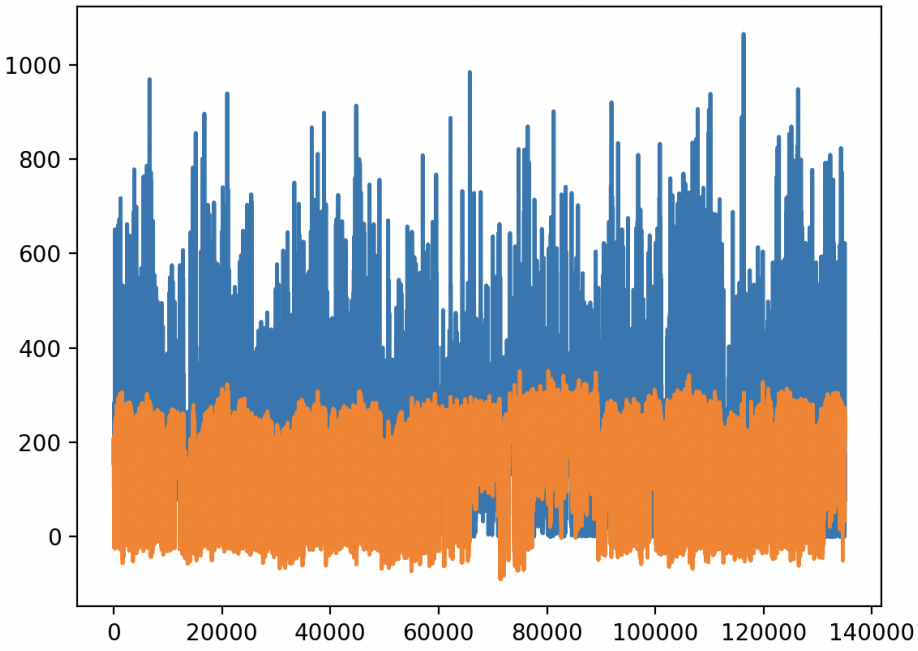
1-Second data set

Linearity



Predictions / Performance

$R^2 = 0.37$

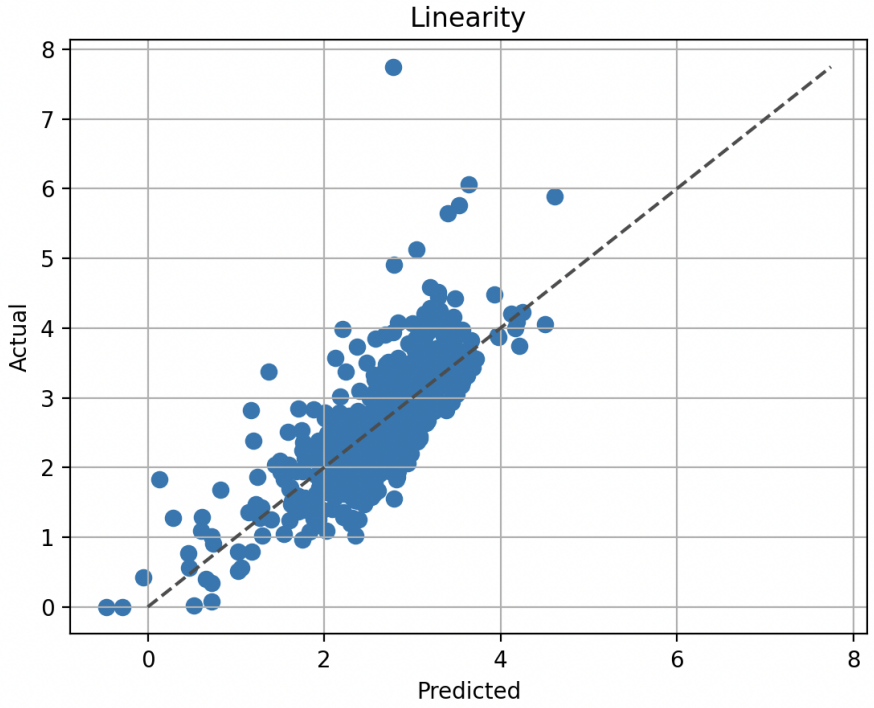


During the process of running the models, it became clear that using power wattage as opposed to power per kilogram was going to be problematic - using a weight-standardised metric was going to be more effective. Moreover, hyperparameter tuning was employed.

After tuning Elastic Net Hyperparameters Using ElasticNet

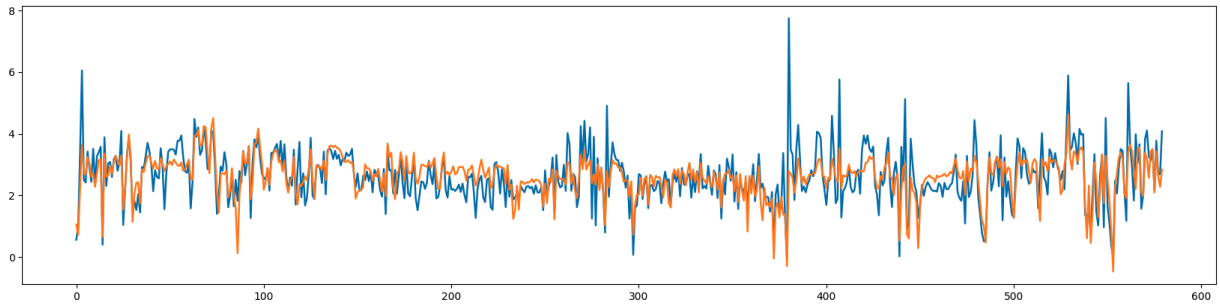
1 minute (smaller) data set- After Hyperparameters tuning

Watt per kilogram



Predictions / Performance

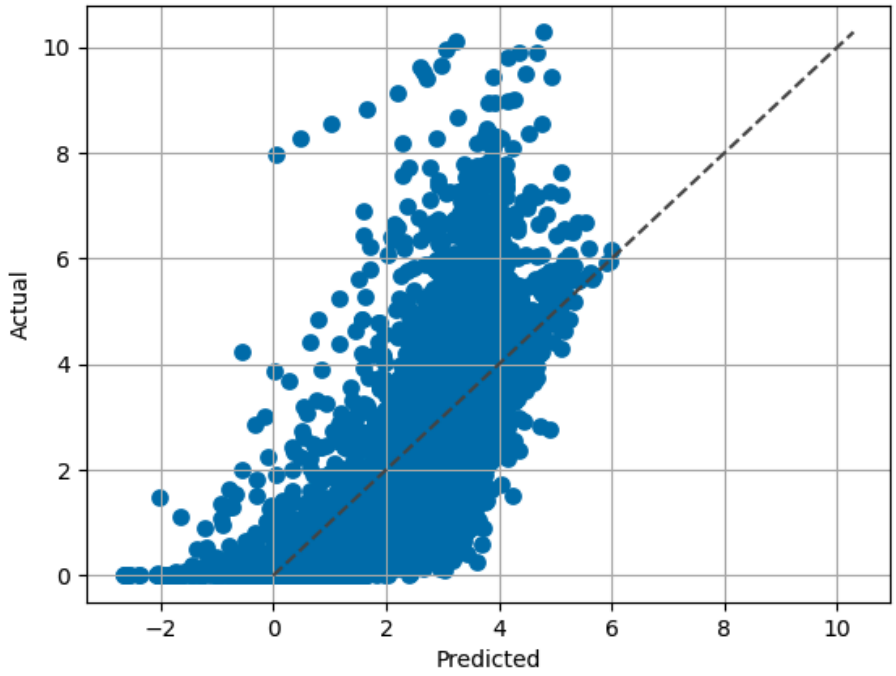
$R^2 = 0.57$



After tuning Elastic Net Hyperparameters Using ElasticNet

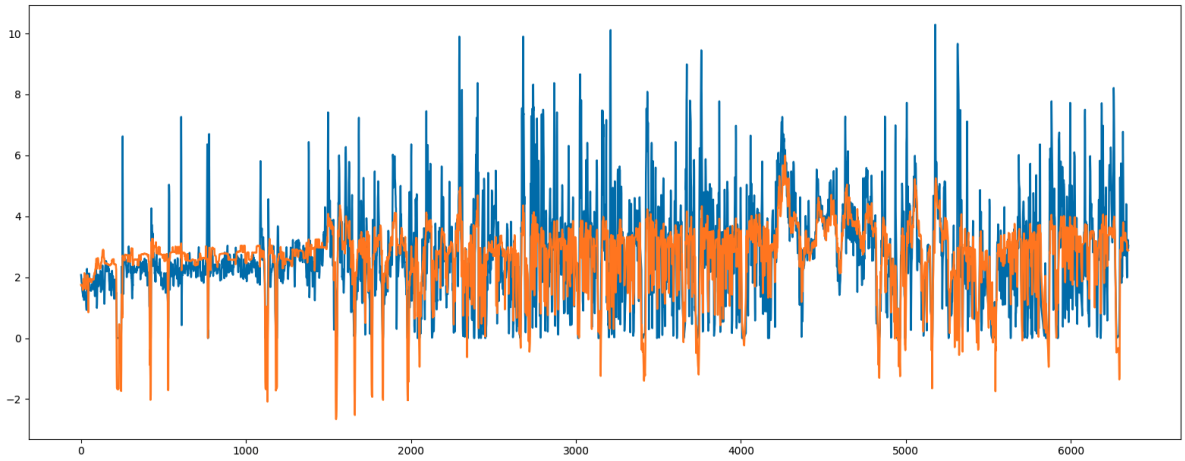
1 minute (larger) data set- After Hyperparameters tuning

Watt per kilogram



Predictions / Performance

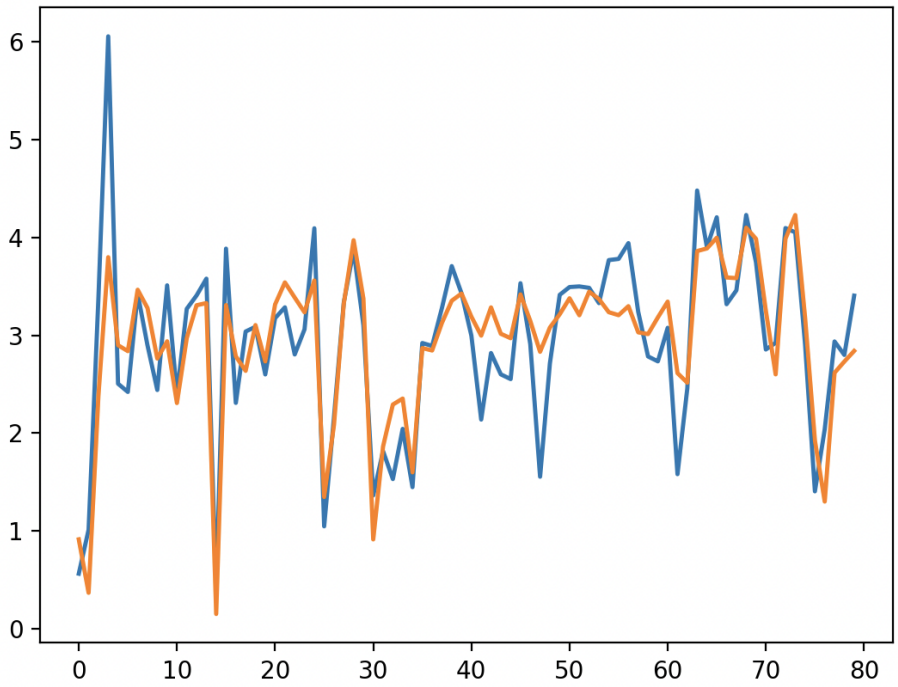
R^2 = 0.47 (overfitting a problem again…)



Summary

Whilst ElasticNet increased accuracy after hyperparameter tuning, it still failed when the amount of data (the number of rows) increased. Further work could be conducted to optimise this implementation of ElasticNet; however, the adoption of such an algorithm is limited due to the dynamics of the problem we are looking to solve. As a further extension and as sklearn's algorithm cheat sheet says, Stochastic Gradient Descent Regressor could be assessed when the number of rows in a given data set exceeds 100k.

It is clear that ElasticNet might be effective when the number of rows/observations is limited to max 100 ($R^2 > 0.75$) see below:



Resources: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html
Resources: https://machinelearningmastery.com/elastic-net-regression-in-python/

Scikit-learn Random Forest - RandomForestRegressor

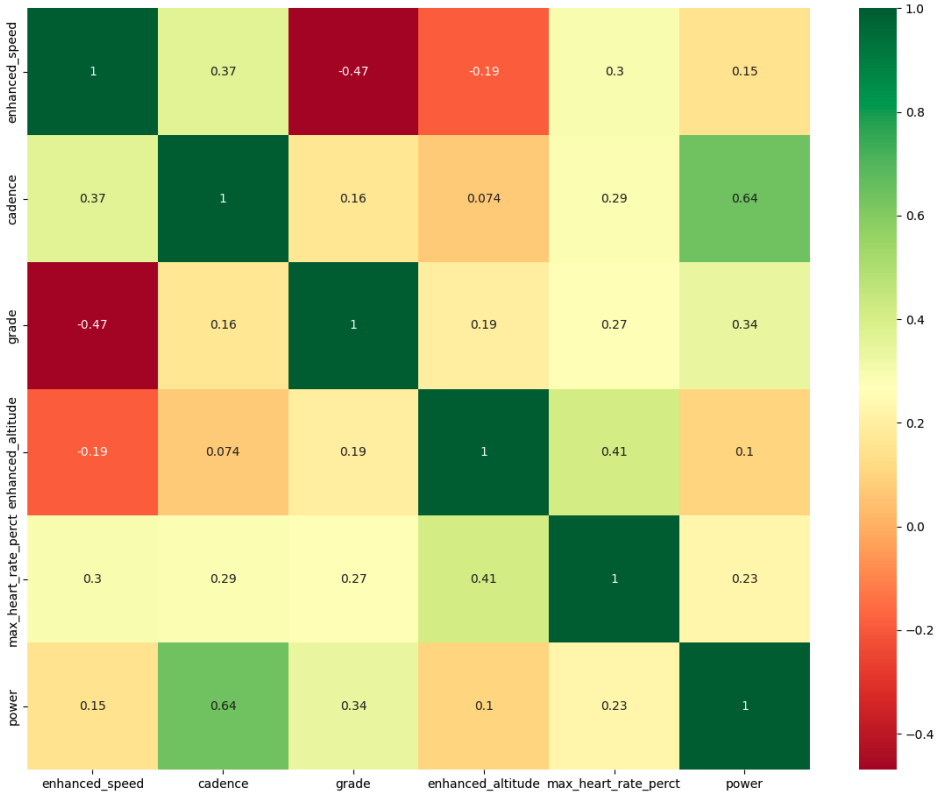
Random forest is a supervised machine learning algorithm that is constructed from multiple decision trees to create a “forest.” It can be used for both classification and regression problems in R and Python. Based on previous results listed further above, Random forest has a default ability to correct for decision trees’ habit of overfitting to their training set - This should of course correct a lot of the issues previous models have faced.

Slightly different data sets have been used for the Random Forest implementation > Additional features have been added to the dataset however the length (number of rows) is similar to the Elastic net dataset (n=6350) where the time interval is in seconds and data is derived from a single user.

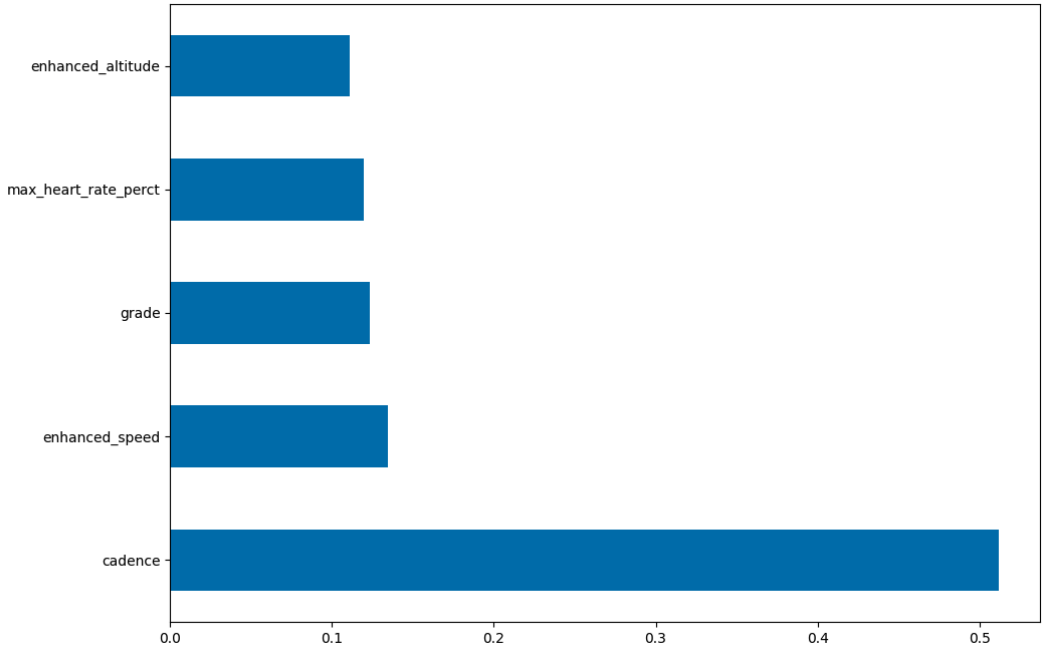
Additionally, a user is offered a choice to either use a short dataset (n=1000) where the time interval is in seconds, a single session dataset (n=112) where the time interval is in minutes or another user’s (User 2) data (n= 8,238)

Dataset 1 (n = 6,350)

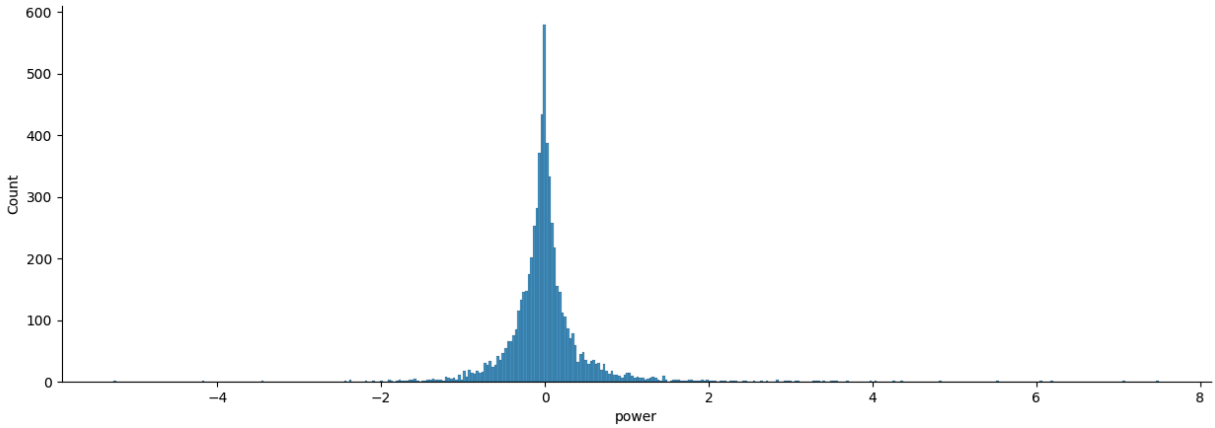
Correlation Matrix:



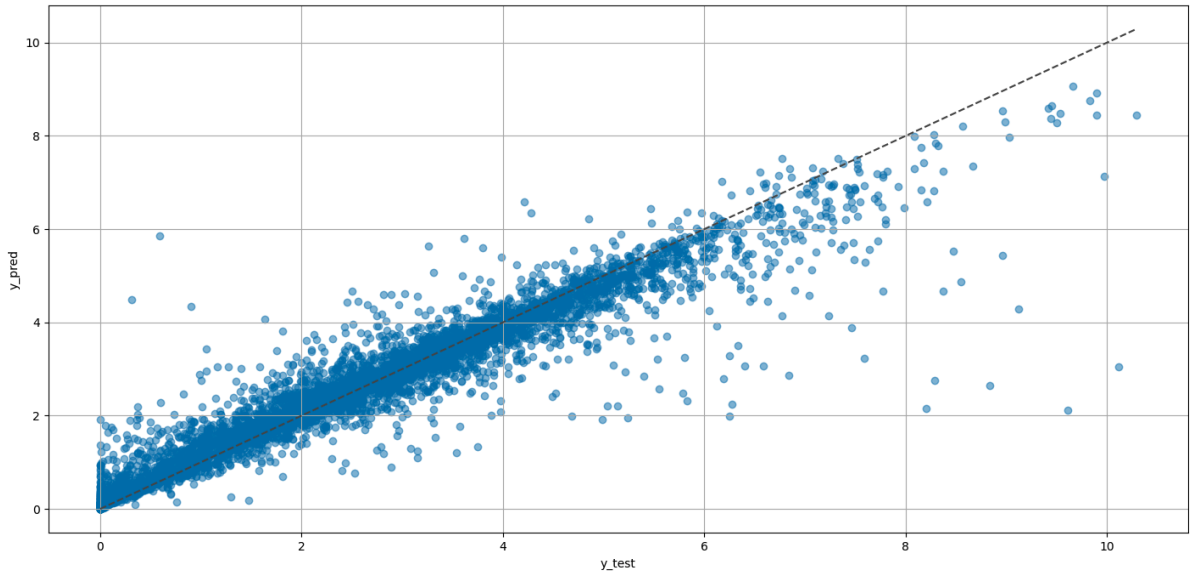
Feature Analysis (Importance):



Distribution Plot (Check model performance)



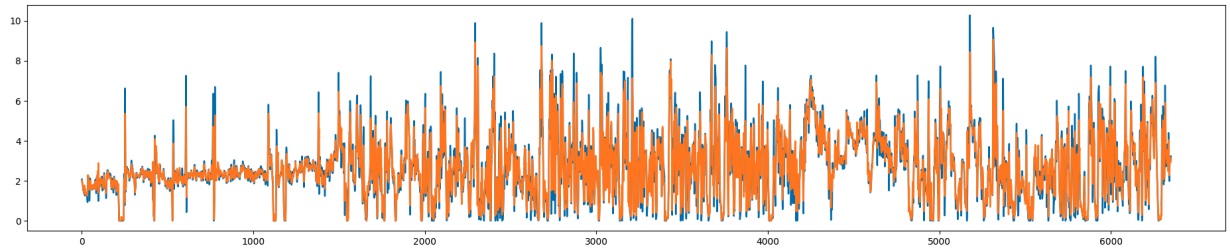
Scatter Plot: Prediction vs Actual



Predictions / Performance (R^2)

Accuracy for training dataset: 0.96%
Accuracy for test dataset: 0.69%

Production
Accuracy for production: 0.90%
MAE: 0.26
MSE: 0.27
RMSE: 0.52



Hyperparameter Tuning - Randomized Search CV

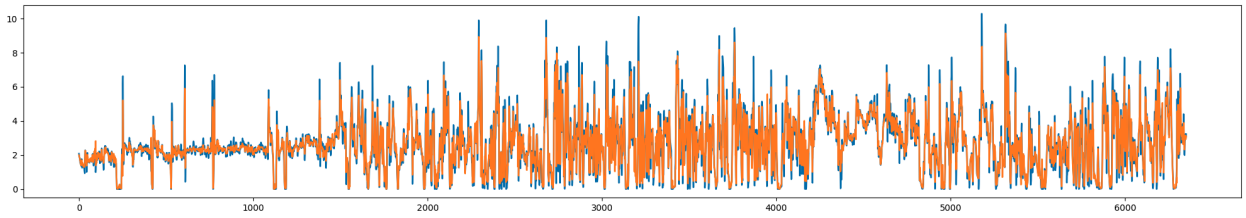
There are two techniques of Hyperparameter tuning i.e
1) RandomizedSearchCv
2) GridSearchCV
We use RandomizedSearchCv because it is much faster than GridSearchCV Number of trees in random forest

Dataset 1 (n = 6,350) after hyperparameter tuning:

Predictions / Performance (R^2)

A minor ~1% increase in performance

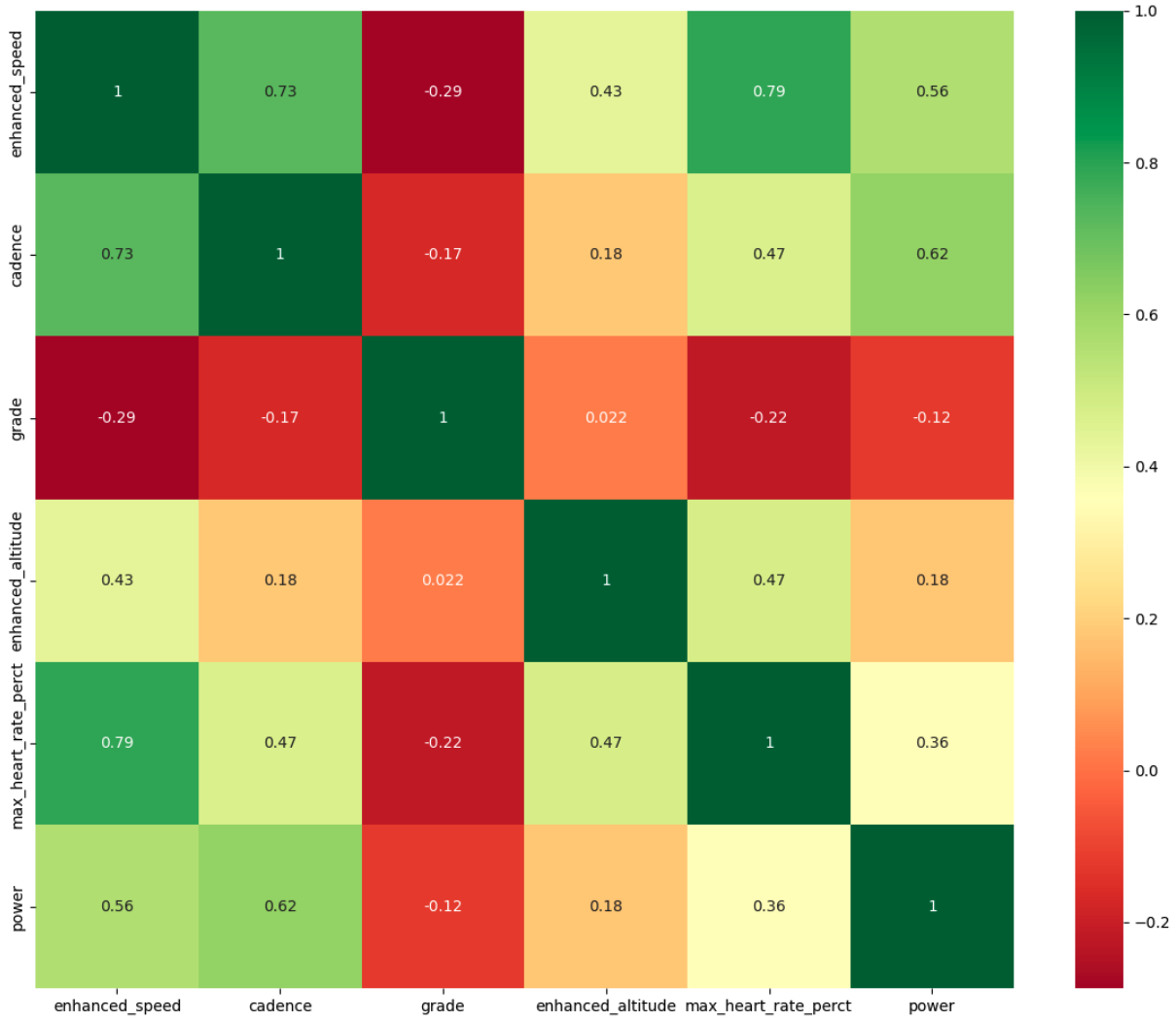
Production
R^2: 0.91
MAE: 0.28
MSE: 0.25
RMSE: 0.50



Dataset 1 (n = 1,000)

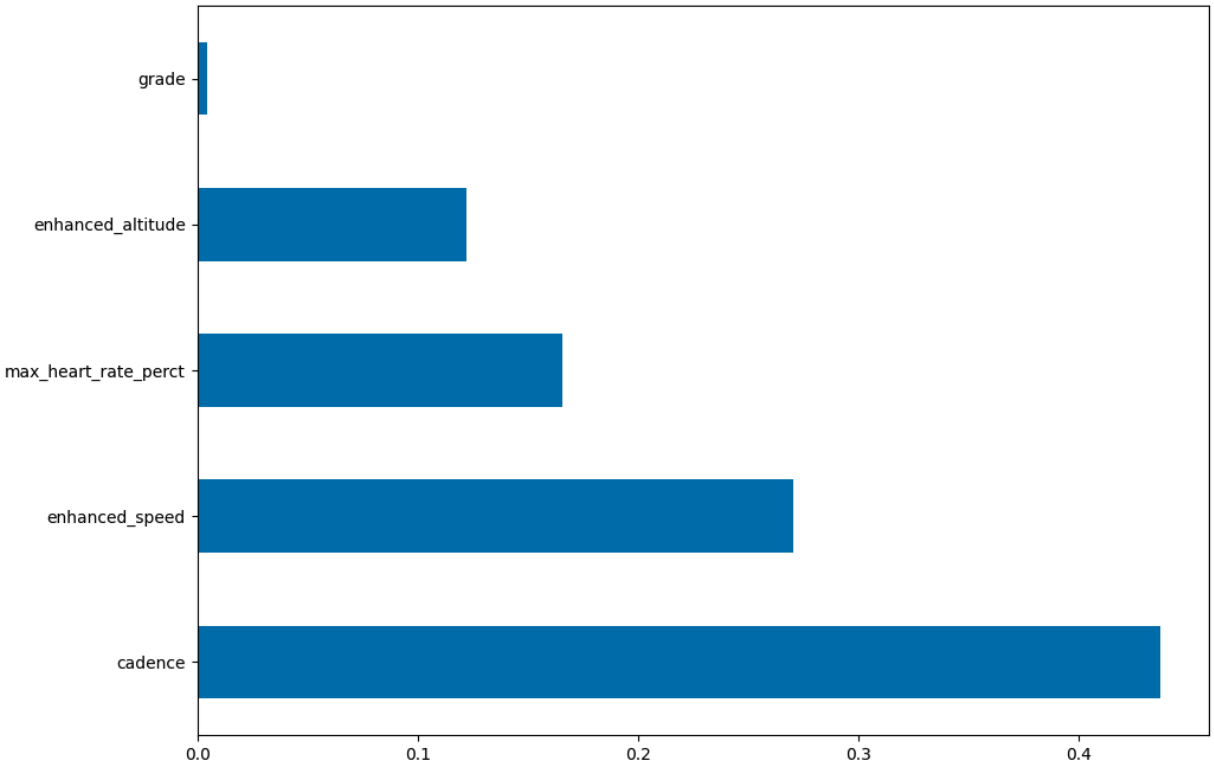
Correlation Matrix:

Strong correlations are not present vs larger data set.

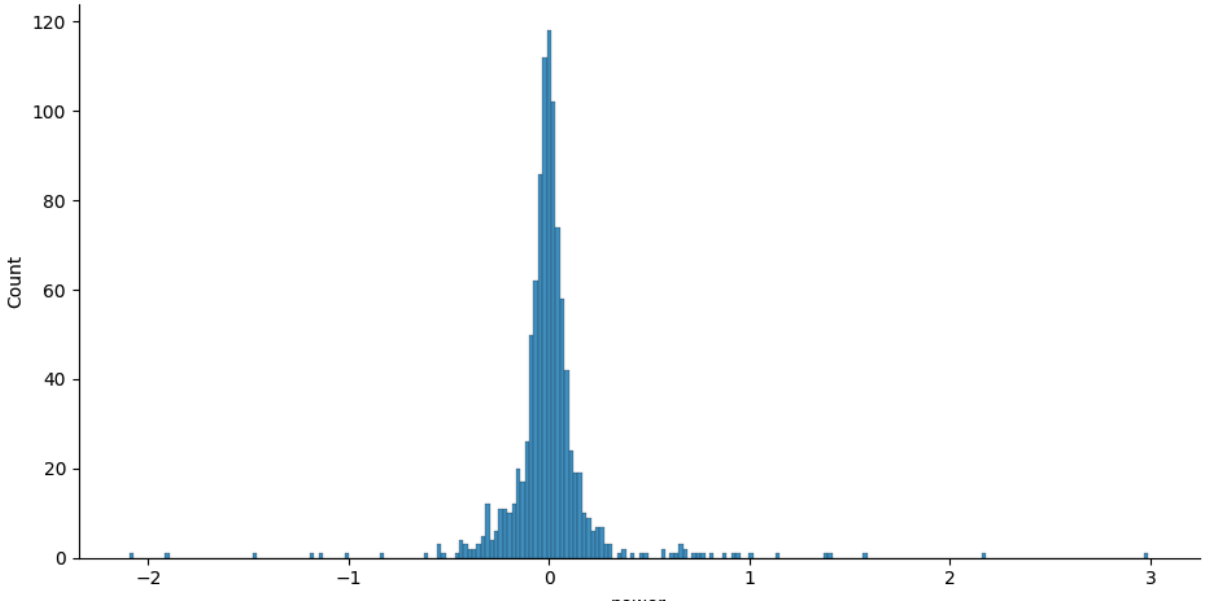


Feature Analysis (Importance):

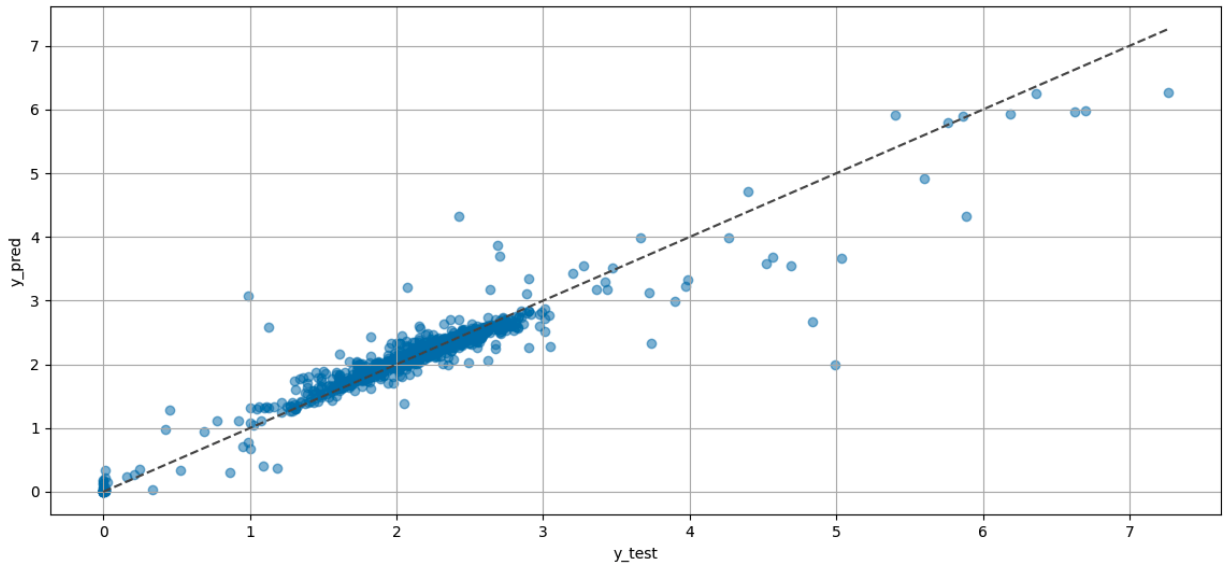
A more distributed shared of feature importance



Distribution Plot (Check model performance)



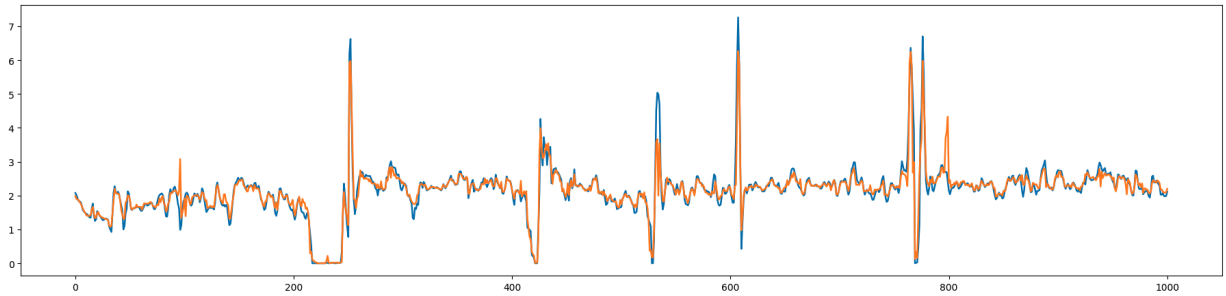
Scatter Plot: Prediction vs Actual



Predictions / Performance (R^2)

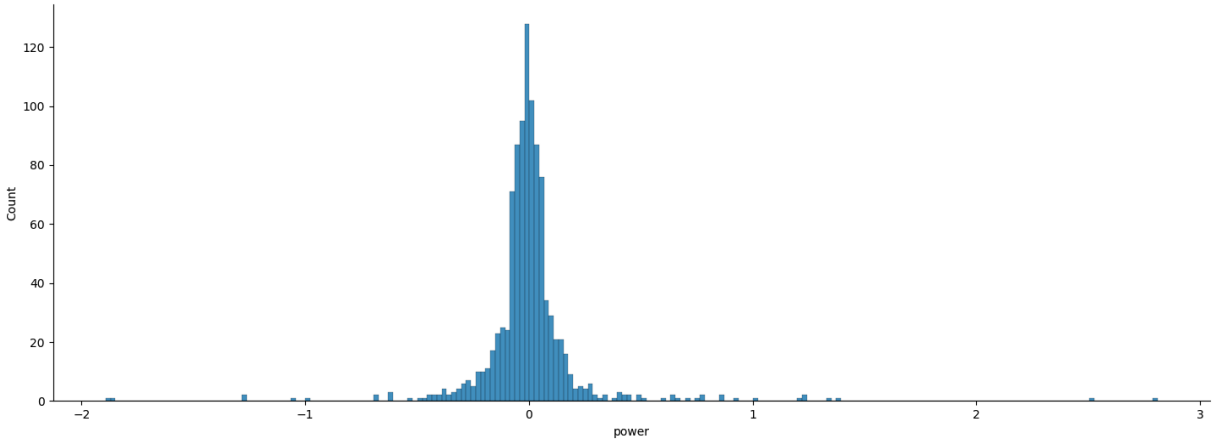
Accuracy for training dataset: 0.95
Accuracy for test dataset: 0.61
Accuracy for production: 0.89

Production
R^2: 0.89
MAE: 0.12
MSE: 0.06
RMSE: 0.25

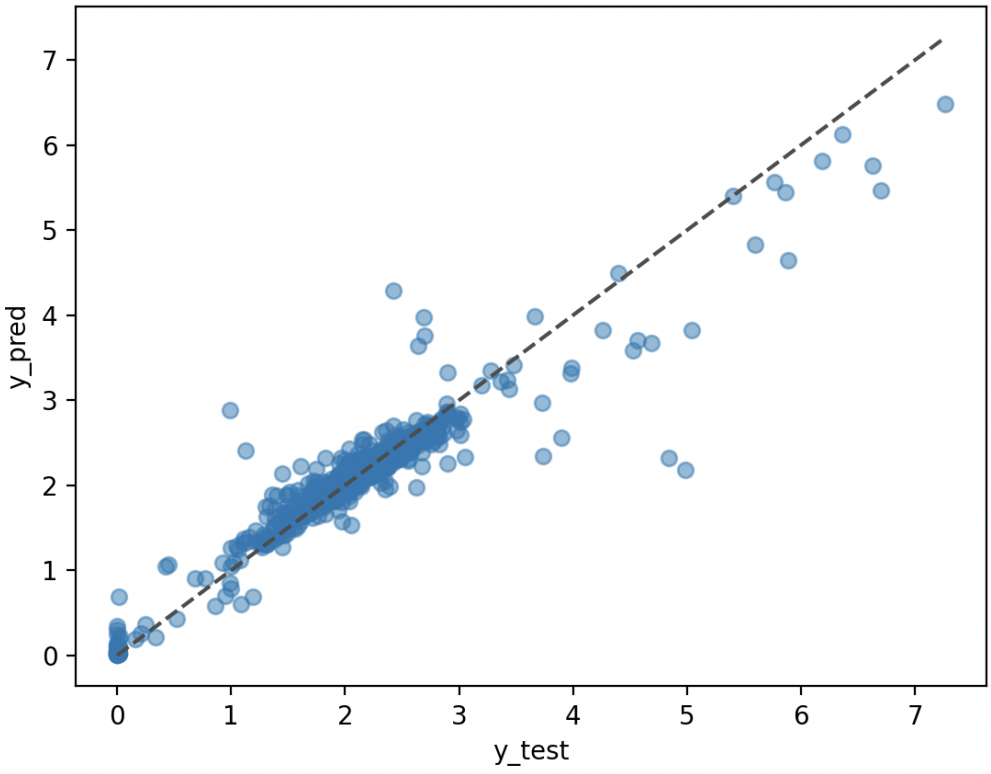


Dataset 1 (n = 1,000) - Shortened, after hyperparameter tuning:

Distribution Plot (Check model performance)



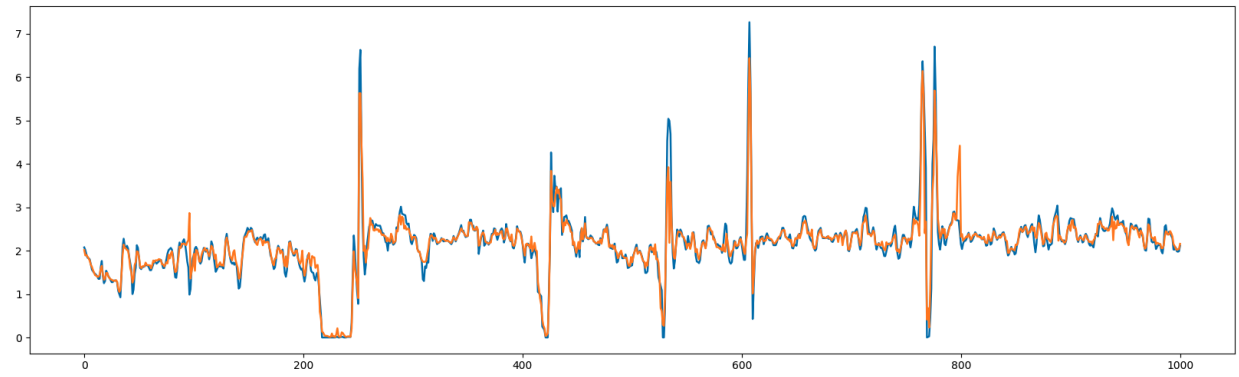
Scatter Plot: Prediction vs Actual



Predictions / Performance (R^2)

Slight improvement after tuning:

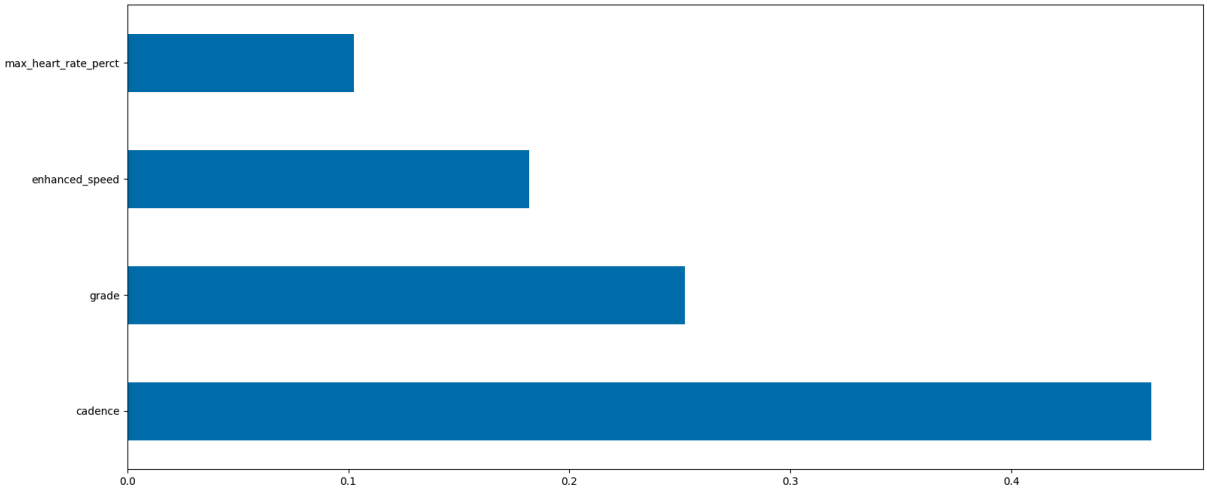
Production
R^2: 0.90
MAE: 0.11
MSE: 0.059
RMSE: 0.24



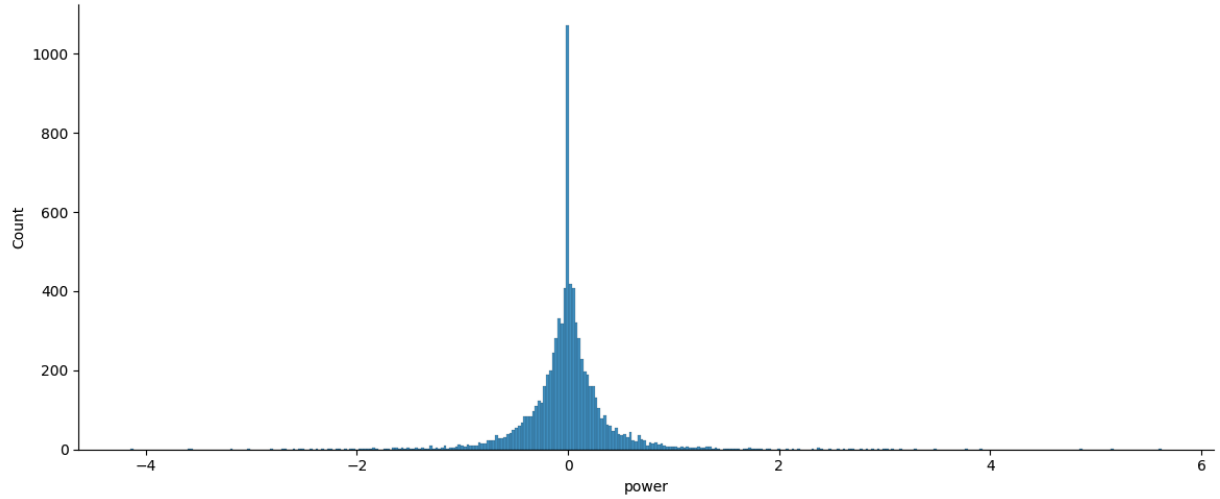
Dataset 2 (n =8,238)

Correlation Matrix:

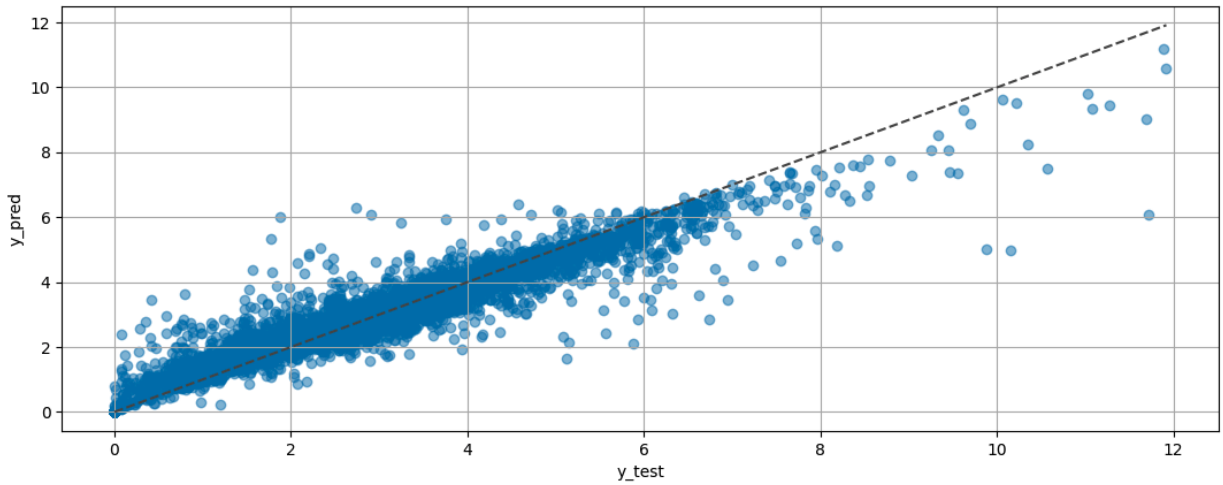
Feature Analysis (Importance):



Distribution Plot (Check model performance)



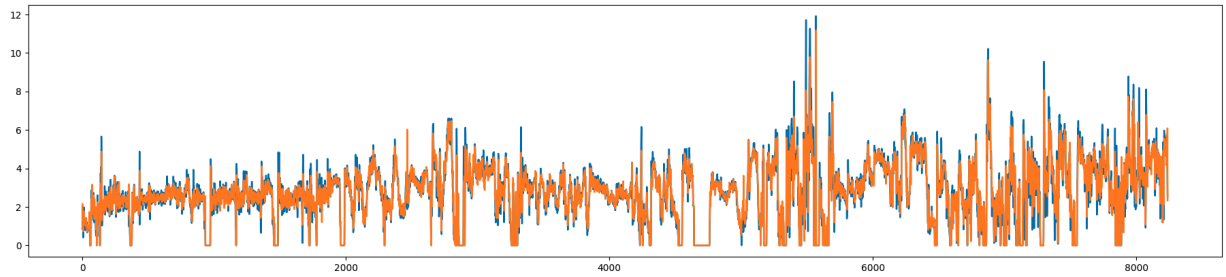
Scatter Plot: Prediction vs Actual



Predictions / Performance (R^2)

Accuracy for training dataset: 0.96
Accuracy for test dataset: 0.73
Accuracy for production: 0.92

Production
R^2: 0.92
MAE: 0.25
MSE: 0.20
RMSE: 0.44

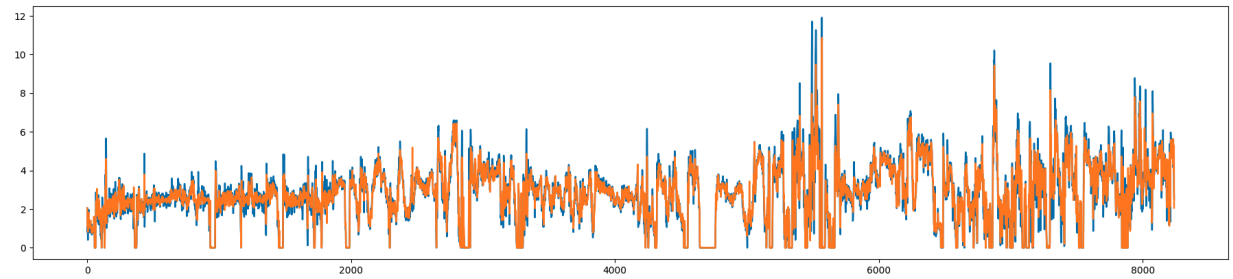


Dataset 2 (n =8,238) After hyperparameter tuning:

Predictions / Performance (R^2)

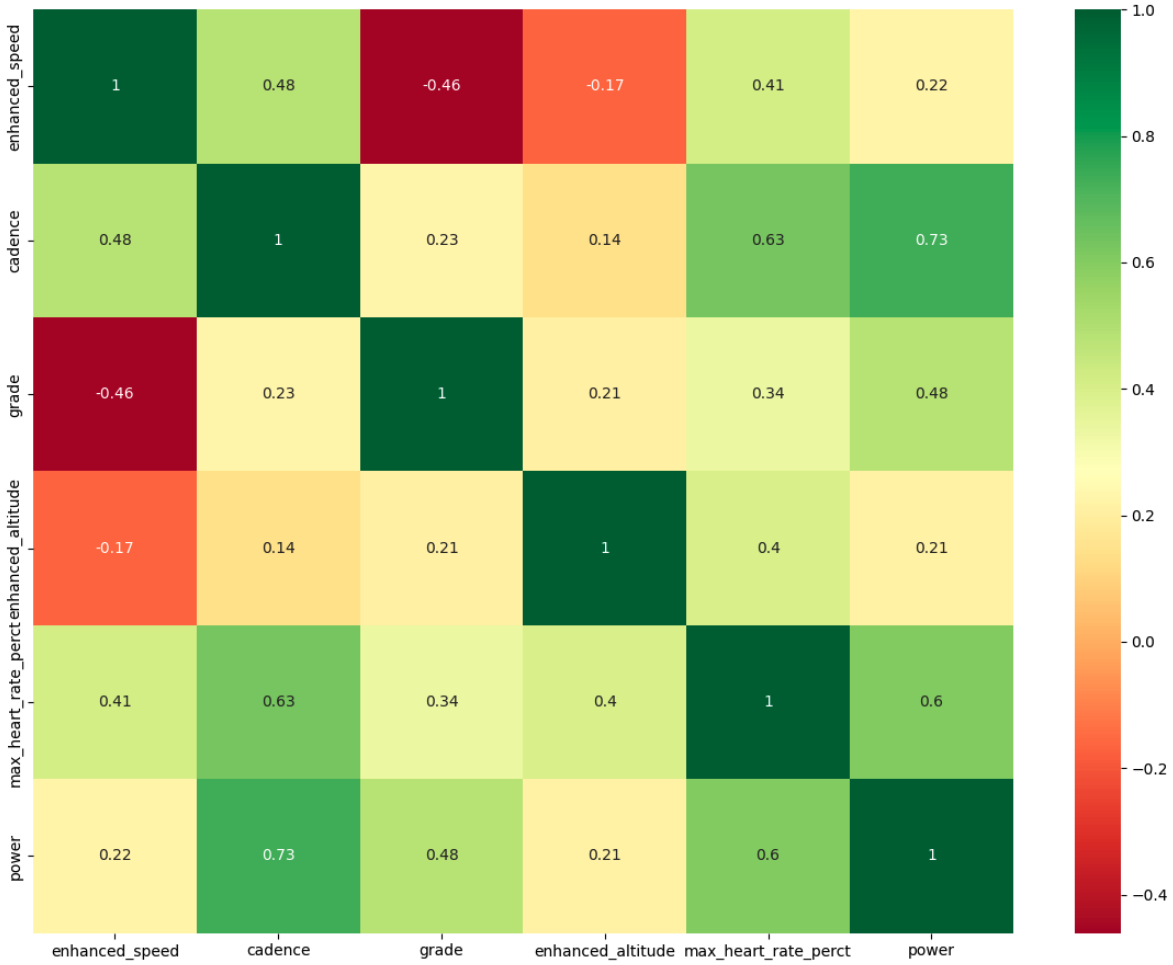
A minor ~1% increase in performance

Production
R^2: 0.92
MAE: 0.25
MSE: 0.19
RMSE: 0.43

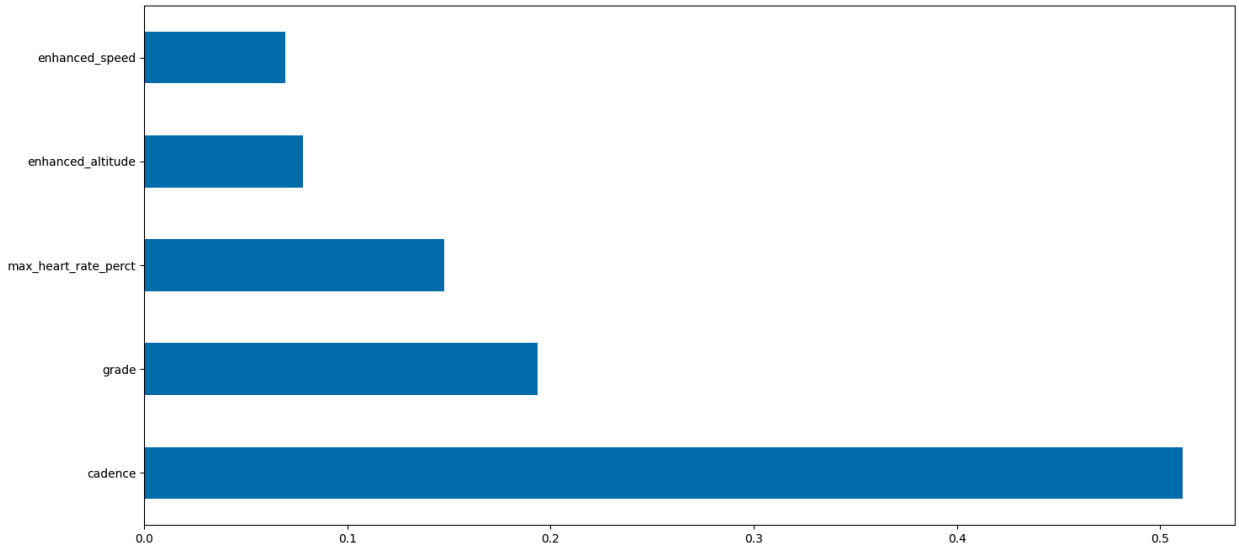


Dataset 3 (n = 112)

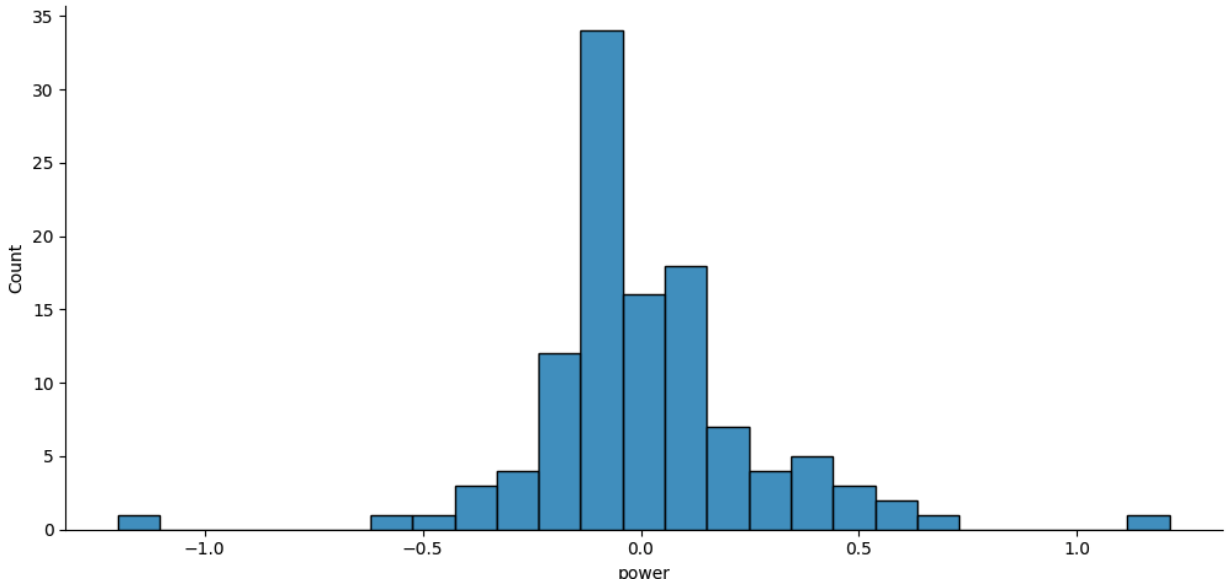
Correlation Matrix:



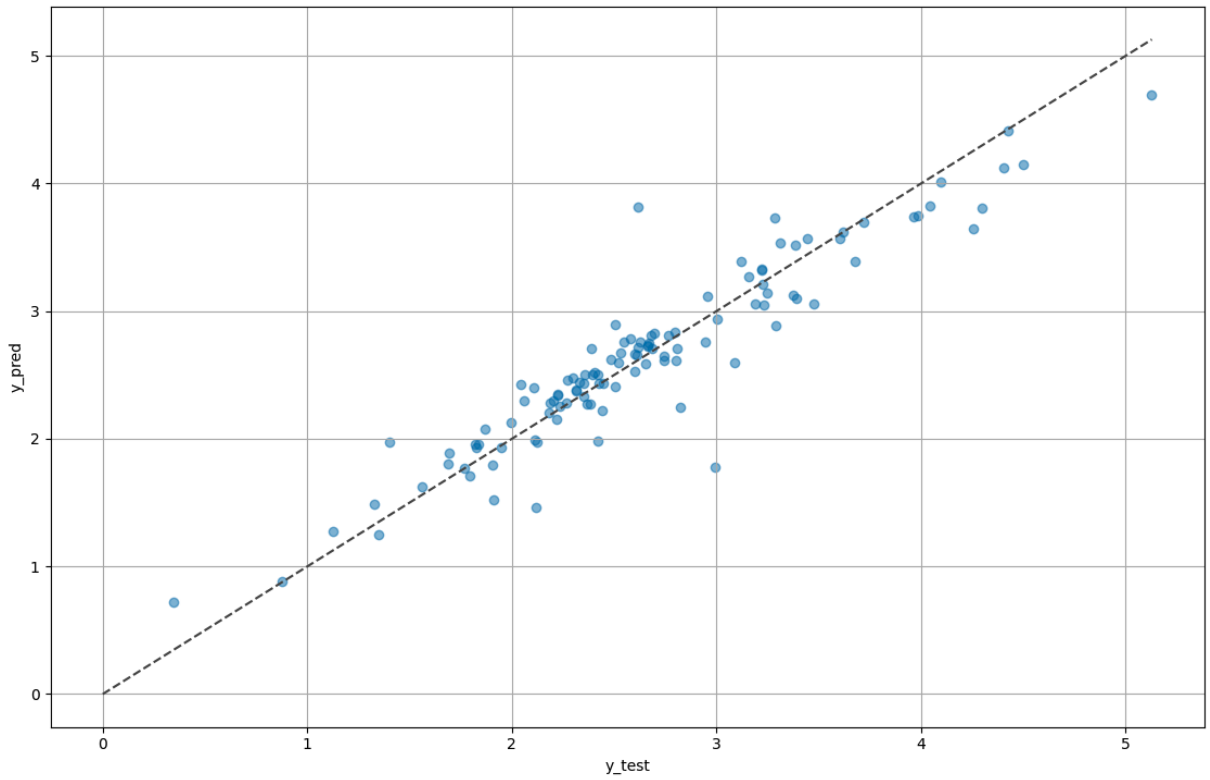
Feature Analysis (Importance)



Distribution Plot (Check model performance)



Scatter Plot: Prediction vs Actual

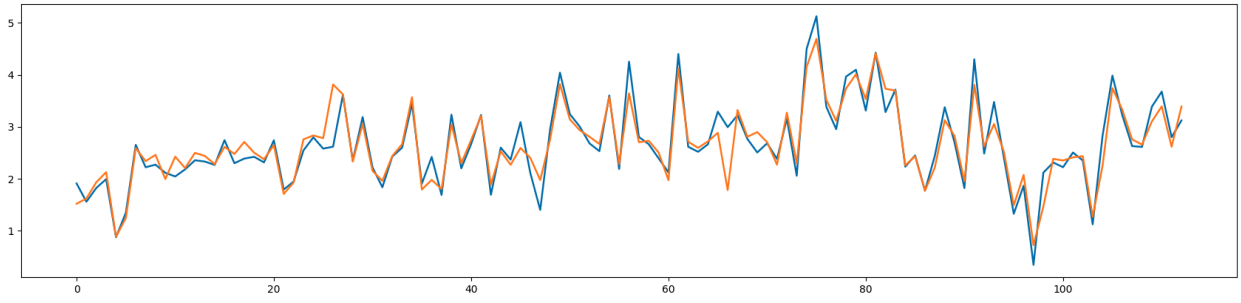


Predictions / Performance (R^2)

Accuracy for training dataset: 0.94
Accuracy for test dataset: 0.63
Accuracy for production: 0.88

Production

R^2: 0.88
MAE: 0.18
MSE: 0.074
RMSE: 0.27

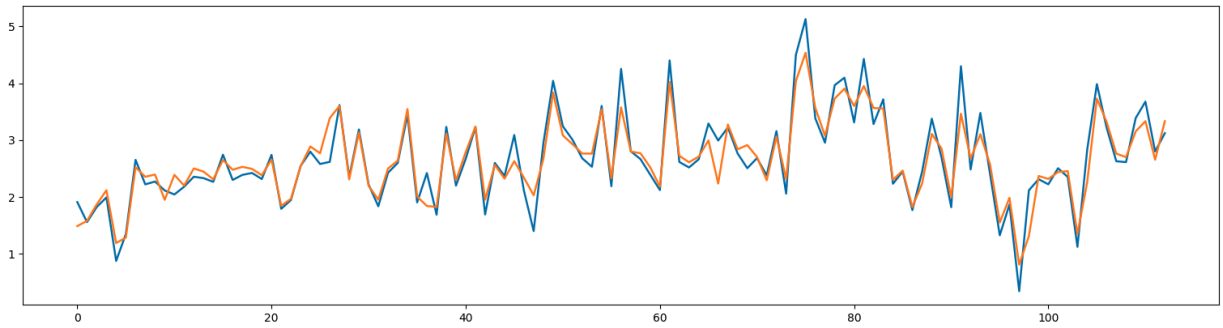


Dataset 3 (n = 112) After hyperparameter tuning:

Predictions / Performance (R^2)

A minor ~1% increase in performance

Production
R^2: 0.88
MAE: 0.19
MSE: 0.072
RMSE: 0.26

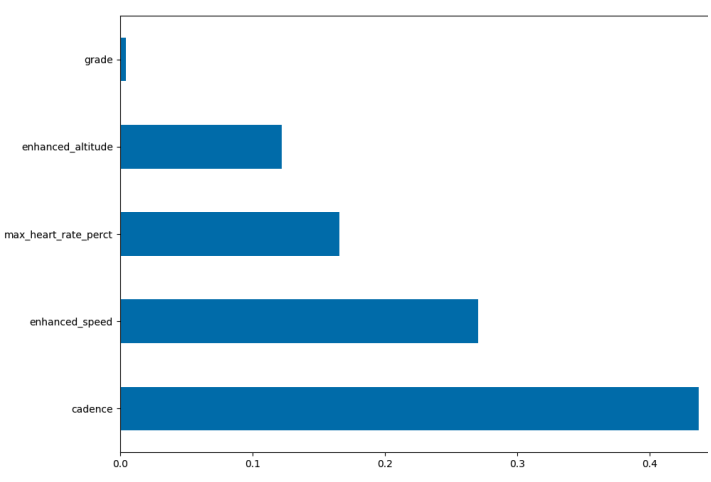


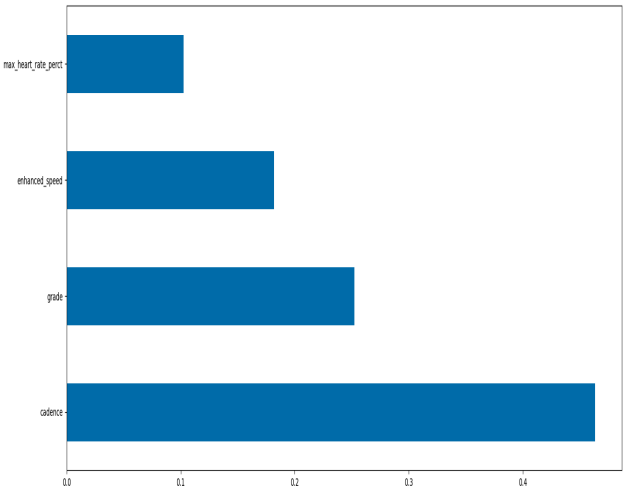
Summary of Results

Model performance was optimal when nrows >= 1000, and the time interval was set as seconds. Moreover, MAE was optimal when the dataset was 1,000 rows; There is a balance in how much data can be used; too little and accuracy suffers, too much and the gap between predicted and actual begins to open. Therefore, further research and effort need to be invested in pursuing models such as LSTM or ANN (deep learning).

Dataset	User 1	User 1 Short	User 2	Session 131
Nrows	6,350	1,000	8,238	112
R^2	0.91	0.91	0.92	0.88
MAE	0.28	0.11	0.25	0.19
MSE	0.25	0.059	0.19	0.072
RMSE	0.5	0.24	0.43	0.26

Interestingly, feature ranking did differ from User (User 1 left) to User (User 2 right): Both had cadence as the key feature however grade was ranked last in one instance vs second in another. This shows how random forest works to weigh features - Both models achieved similar performance levels.





Conclusion

To ensure some consistency across all efforts, the Session 131 dataset was used across all three models, results below; As expected Random Forest was the strongest performer; this comes to problem statement and model sit - Random Forest met the basic requirements for being able to effectively predict power out for a given Cyclist's workout session using a range of other cycling performance metrics.