Final Proj: UAV BLADE GEOMETRY

# Capstone proj: Col Rakesh Pedram 19 Feb 23

## Brief Obj

Develop a data-driven framework for enhancing the prediction of propeller performance, crucial for the evolving field of urban air mobility. Utilize modern research trends, incorporating both standalone and ensemble machine learning techniques to forecast thrust coefficient, power coefficient, and efficiency based on blade geometry and operational inputs.

### Research for domain knowledge

What are Python naming conventions?

Python has specific **naming conventions** for variables, functions, classes, and other entities in your code. Adhering to these conventions ensures consistency, readability, and better collaboration among developers.:

1. **Case Sensitivity**:
   * Python identifiers are **case-sensitive**. For example, myVariable and myvariable are considered different names.
   * Choose a consistent case style (either **snake\_case** or **CamelCase**) and stick to it throughout your code.
2. **Variable Names**:
   * Variables should have descriptive names that convey their purpose.
   * Use **snake\_case** for variable names (all lowercase letters with underscores between words). For example:
   * user\_age = 30
   * total\_items = 100
3. **Function Names**:
   * Functions should also use **snake\_case**.
   * Choose descriptive names that indicate the function’s purpose.
   * For example:
   * def calculate\_total(items):
   * # Function logic here
   * pass
4. **Class Names**:
   * Class names should use **CamelCase** (capitalize the first letter of each word, without spaces or underscores).
   * For example:
   * class MyClass:
   * # Class definition here
   * pass
5. **Constants**:
   * Constants (values that don’t change) should be in **UPPERCASE** with underscores separating words.
   * For example:
   * MAX\_ITEMS = 100
   * PI = 3.14
6. **Avoid Single-Character Names**:
   * Unless the variable represents an iterator (e.g., i, j, k), avoid using single-letter names.
   * Instead, choose more descriptive names.
7. **Avoid Reserved Words**:
   * Do not use Python’s reserved words (e.g., if, else, while, for, etc.) as identifiers.

Remember, **readability counts**! Consistency within your project is crucial, and adhering to these conventions will make your code more maintainable and understandable. If you have any doubts, feel free to ask! 🐍

Research was carried out on the fluid dynamics behind the blade design to get more insights into domain knowledge

1. **Blade Geometry of a Propeller**:
   * A propeller blade is a rotating airfoil that produces lift and drag. Each blade has several key features:
     + **Diameter**: The diameter is the imaginary circle scribed by the blade tips as the propeller rotates.
     + **Radius**: The distance from the axis of rotation to the blade tip (radius multiplied by two equals the diameter).
     + **Blade Face**: The aft side of the blade (surface facing the stern).
     + **Blade Back**: The forward side of the blade (surface facing the bow).
     + **Leading Edge (LE)**: The edge of the propeller blade adjacent to the forward end of the hub. It leads into the flow when providing forward thrust.
     + **Trailing Edge (TE)**: The edge of the propeller adjacent to the aft end of the hub. It retreats from the flow when providing forward thrust.
     + **Blade Number**: Equal to the number of blades on the propeller.
     + **Blade Tip**: The maximum reach of the blade from the center of the hub.
     + **Blade Root**: Separates the leading and trailing edges.
   * [The blade sections have varying values of local face pitch on the pitch side or blade face 1](http://navalex.com/downloads/Michigan_Wheel_Propeller_Geometry.pdf).
2. **Solidity**:
   * Solidity is a dimensionless quantity used in the design and analysis of rotorcraft, propellers, and wind turbines.
   * It is the ratio of the area of the rotor blades to the area of the rotor disk.
   * Solidity provides a measure of how close a lifting rotor system is to an ideal actuator disk in momentum theory.
   * It plays a crucial role in determining the fluid speed across the rotor disk when lift is generated, affecting rotor performance, downwash, and noise levels.
   * [Typical values of rotor solidity for helicopters fall in the range of 0.05 to 0.12](http://navalex.com/downloads/Michigan_Wheel_Propeller_Geometry.pdf) [2](https://en.wikipedia.org/wiki/Rotor_solidity).
3. **Efficiency of a Propeller**:
   * Propeller efficiency is a measure of how effectively a propeller converts engine power into useful thrust.
   * The ideal efficiency of any propulsor is that of an actuator disk in an ideal fluid (called the Froude efficiency).
   * In practice, propeller efficiency typically peaks around 0.8 before various aerodynamic effects decay its performance.
   * Efficiency depends on factors like blade design, solidity, and operating conditions.
   * [Propeller efficiency can be calculated using parameters such as thrust coefficient, torque coefficient, and power coefficient](http://navalex.com/downloads/Michigan_Wheel_Propeller_Geometry.pdf) [3](https://web.mit.edu/16.unified/www/FALL/thermodynamics/notes/node86.html).
4. **Geometry Refinement**:
   * To refine propeller geometry, designers optimize blade shape, twist, and airfoil sections.
   * Computational methods, wind tunnel testing, and flight testing help refine blade profiles.
   * Iterative design processes aim to improve efficiency, reduce noise, and enhance performance.

## Execution of project work

### Project brief

The key aspects of the project brief were studied and summerised as under

* We have to design a data model that predicts the propeller efficiency based on the geometry and wind tunnel experiments carried out.
* Upload the data od experiments on a Mysql server and execute some queries to get the outputs
* Make a visualization in tableau for the company

### Trapz() Function

I missed this one in the first go, wasting hours in debugging. Should have read the cook book.

The function returns area with the chord and radius fed in as arrays and then integrated. Should have understood this function better as I was not feeding it an array filtered by the blade name. Very sketchy info available online the brief could have made it more clear.

## Data wrangling

The data set was examined after importing into python. I used google colabs for the project. All the files were concatenated into two set, one for experiment and the other for the geometry.

The following key issues were noticed.

* Not all the blades in the experiments have geometry defined in the geometry dataset. Hence once cannot compute their blade area and solidity which are the key factors in the model. These are almost 40% of the dataset hence applying any missing value algorithm would defeat the purpose.
* I discovered to drop these from the model. Only discovered this anomaly after painful debugging where I was getting null values in my solidity and blade area algo.
* There are some -ve values in efficiency. Which is illogical and these outliers had to be removed.
* Applied the descriptive statistics on the data set.
* Based on these I concluded:-

I had to normalize the data for the following fields as their ranges are not comparative and will lead to a magnitide bias in the model. 'propellers\_diameter', 'propellers\_pitch',

       'advanced\_ratio\_input', 'rpm\_rotation\_input',

       'thrust\_coefficient\_output', 'power\_coefficient\_output',

       'efficiency\_output', 'blade\_area', 'disc\_area', 'solidity'

* + I used the MinMaxScalar() from sklearn library.

### Correlation matrix

I ran the correlation matrix on the normalized data using seaborn the plot is below

A screenshot of a computer screen

Description automatically generated

I dropped the power\_coeff and thrust coeff variables from the matrix as they showed poor correlation. This was non intuitive as they are major factors in a efficiency equation but I decided to go with the results and reduce my dimensionality.

I studies various scatter plots to understand the correlation between factors. Most correlation coeff were not very strong statistically but logistic regression was considered as a model to incorporate multiple variables.

A blue and white chart

Description automatically generated

A graph of a diagram

Description automatically generated with medium confidence

A blue dotted diagram with white text

Description automatically generated

## Model Building

The target variable was efficiency and the independent variables were: all the normalized less the variables dropped which was already discussed.

I made three modela using gradient boosting aggressor :

* + Without imputation
  + With Imputation
  + By deleting solidity

The models were all trained on the 2 blade data and predictions applied on more than two blade data.

Sample code :-

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import GradientBoostingRegressor

from sklearn.impute import SimpleImputer

from sklearn.metrics import mean\_squared\_error

# ... other imports for evaluation metrics and visualizations

# Preprocessing & Feature/Target Separation

X = df\_filtered.drop('efficiency\_output', axis=1) # Input features

y = df\_filtered['efficiency\_output'] # Target variable

# Train Test Split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Model Building (we have three scenarios) The imputations isnt likely to help

## Model 1: No imputation

model1 = GradientBoostingRegressor()

model1.fit(X\_train, y\_train)

# 6. Evaluation on Propellers with Other Blade Counts

df\_other\_blades = df\_model[df\_model['number\_of\_blades'] != 2]

X\_other = df\_other\_blades.drop('efficiency\_output', axis=1)

Y\_other=df\_other\_blades['efficiency\_output']

# ... Impute if model2 used imputation

# ... Remove solidity column if model3 is used

predictions1 = model1.predict(X\_other)

print('Mean square error of model 1 no imoutation is :',mean\_squared\_error(Y\_other, predictions1))

r\_squared = model1.score(X\_other,Y\_other) # Use your test set

print("R-squared on model 1 No imputation is:", r\_squared)

### Analysis of the result

I graphed the Rsquare and the MSE on graph for all three models

A graph of blue rectangular bars

Description automatically generated with medium confidenceA graph of blue rectangular shapes

Description automatically generated with medium confidence

The model without imputation fitted the best and was accepted. There was very little difference between the model with and without imputation. The model less solidity expectedly performed poorly.

The R Square was more than 0.5 is considered good in conjunction with very low MSE.

There after using the modelled data I plotted the test and predicted values to graphically show the performances of the model

A diagram of blue dots

Description automatically generated

A diagram of blue dots

Description automatically generated

A diagram of blue dots

Description automatically generated

Visually its seen that model was successfully fitted and gave good results. This was anticipated as there was no categoric and ordinal data considered in the fitting and observations have no biases.

As I couldn’t upload my notebook on the sire the script is attached as text.

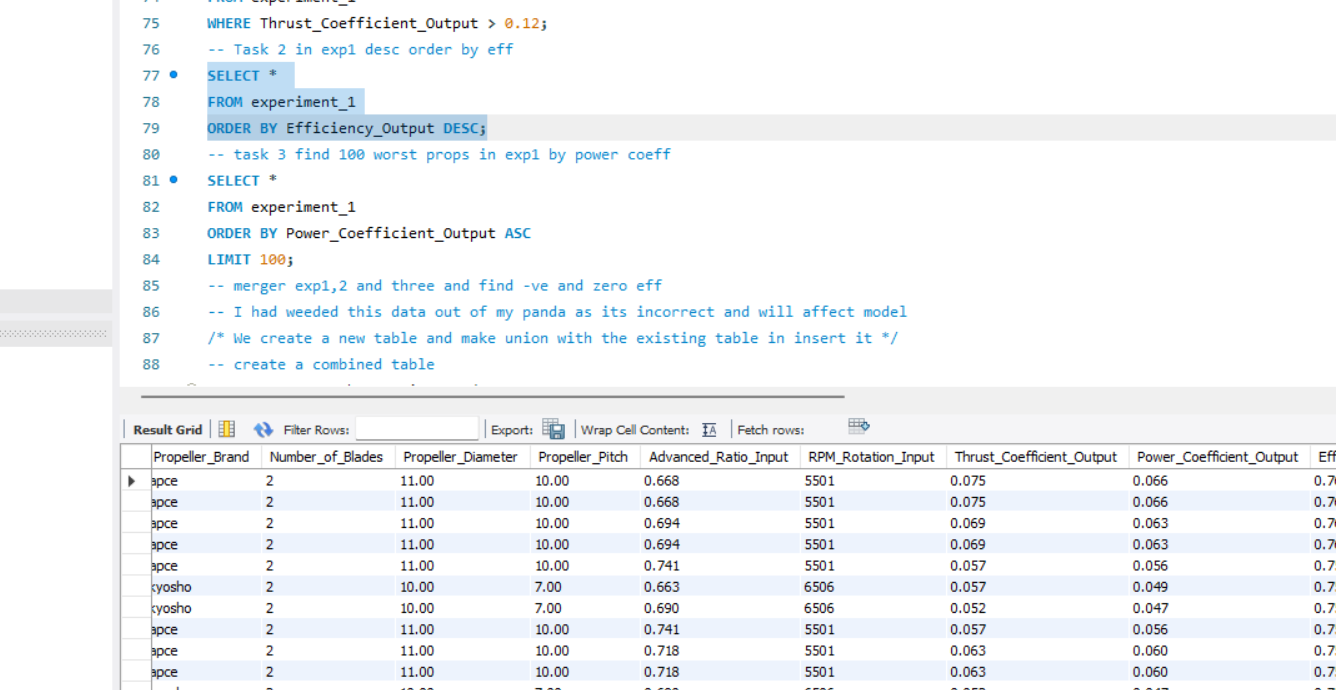
## SQL Portion

### Creation of tables and schema

Based on the existing data in the experiment\_vol1 files. New DB was made and tables were first created in the db.

* The csv files were then loaded in the db. Label matching was ensured along with data type and there were no errors.
* Thereafter simple sql queries were run to fetch the data.
* Finally a new combined table was made and all the other tables unionized in the same and a qury run on this.
* The SQL portion was very basic. Code is below.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

## Tableau Visualisation

I made the following assumption on the companies nature business. I assume the firm is a R&D tests various test data and makes recommendation.

Hence my Tableau visualization is based on the use case of a constructor who is looking for the ideal propeller for his UAV.

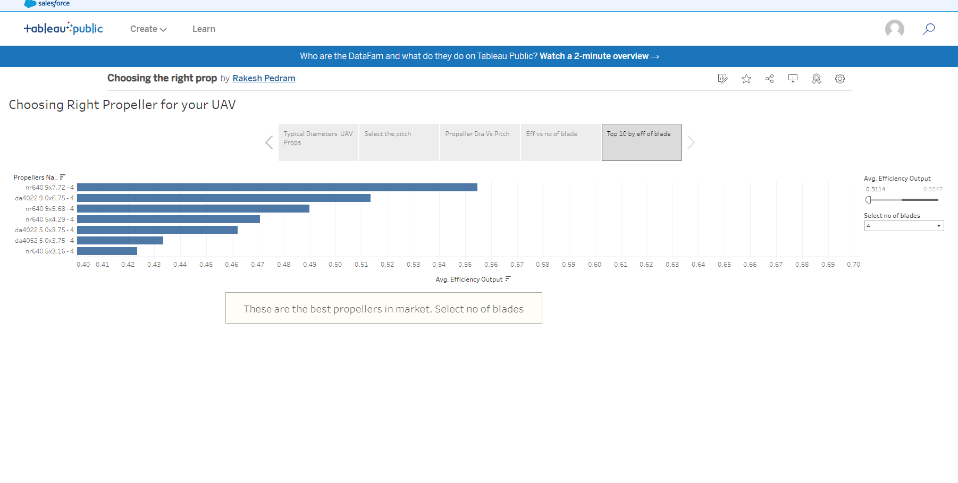
The story board was made giving various visualization of dependencies of various parameters on the propeller efficiencies.

The common propeller configurations available in the market

Finally the best 10 propellers based on efficiency and number of blades.

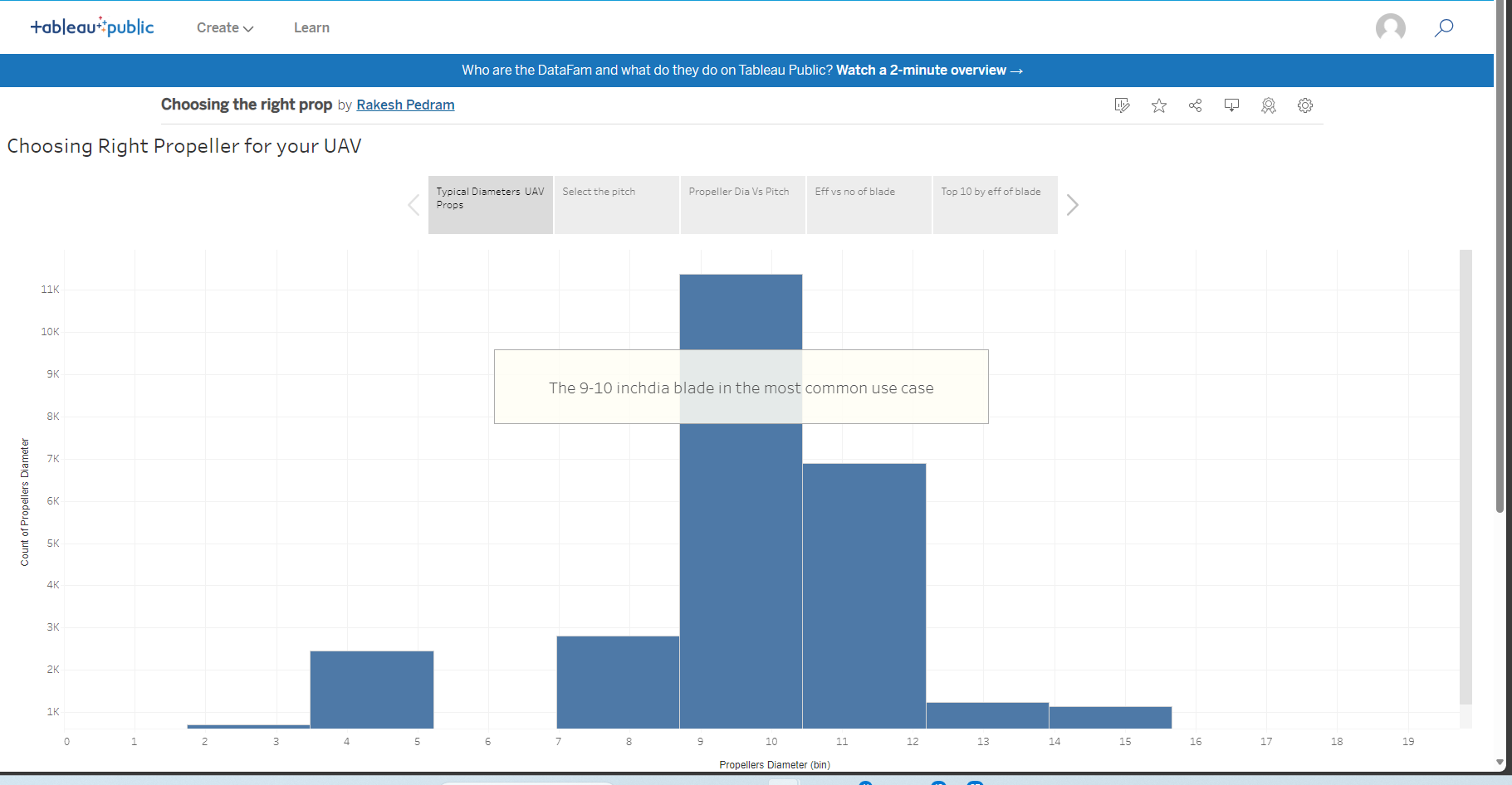
### Execution

I made a parameter for no of blades and this was linked to a calculated field to filter data for various blade configuration. Visualization is uploaded on Tableaus public server. Screen shots are below.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

## Conclusion

The project was tough. I spend a full day in debugging on various errors primarily as I had not cleaned the data and done the initial scrutiny and research well.. The application of the techniques on the data was fully exercised. The data was real life with errors. As this was mathematical the models fitted well.

Skill on choosing and dropping variables in iterative manner was learned well as also implication on normalizing the data.

I enjoyed doing this project was a mighty challenge as I had to learn on my own.

Thanks