

Flight Delay Prediction & Analysis

Odne Rindheim, 10 November 2023

DESCRIBE THE PROBLEM

SCOPE

- *“Business objective”: Describe the project’s goal and “business impact.”*
 - To predict flight delays and understand their primary causes to optimize operations and reduce financial and reputational costs.
- *How will the solution produced in the project be used? What similar solutions already exist / how is the problem solved today? How would you do the task manually without using machine learning?*
 - Airlines can optimize crew schedules, improve customer notifications about delays, and adjust logistics.
 - Currently, airlines use rudimentary predictions based on historical averages and immediate known factors like weather.
 - Without machine learning, one would look at weather forecasts, recent airport congestions, and historical delays for the specific flights.
- *How will the performance be measured via “business metrics”?*
 - Cost savings, reduced number of delays, and improved customer satisfaction scores
- *If your machine learning model will be part of a more extensive “pipeline” or system, describe the system’s components. Consider how changes in one part of the system may impact other parts.*
 - The machine learning model will integrate into a broader flight management system that schedules flights, notifies passengers, and plans crew schedules.
- *Describe the stakeholders of the project.*
 - Airlines management, operational teams, customer/passengers, and airport authorities
- *Describe a tentative timeline for the project. Include milestones.*
 - 2 weeks for data preprocessing and EDA, 2 weeks for model development, 1 week for deployment and continuous monitoring thereafter
- *Define what resources, for example, computational resources and personnel, will be required to complete the project.*
 - High-performance computer cluster for training models, domain experts for feature engineering, and software developers for integration

METRICS

- *Describe the minimal “business metric” performance for the project to be considered a success.*
 - A reduction in costs associated with delays by 10% in the first year. Allow airlines to predict flight delays and set in preventative measures from this affecting crew and passengers.

- *Describe machine learning and software metrics or objectives that will be used to measure whether the system/solution is working. Here, you should use metrics that can be easily calculated. Accuracy, mean squared error, latency, and throughput are examples. Describe how they are connected to the “business objective” you described earlier.*
 - Prediction accuracy, Mean Absolute Error (MSE) for delay duration, system latency, and throughput, these metrics directly impact the ability of efficiently manage flight schedules and customer expectations.

DATA

- *Describe what data and labels you will use. What kind of data is it? Where will you obtain data / how can you collect data? How much data is already available, and how much do you estimate is needed? If the problem is to be attacked using “supervised learning” methods, describe how you will get the “ground truth” labels. How will you ensure that the labels are sufficiently accurate? That they are consistent?*
 - Historical flight data provided by the U.S Department of Transportation’s (DOT) Bureau of Transportation Statistics (BTS).
 - Flight data is already collected and publicly available.
 - The data set contains labels for delay duration and reasons.
- *Describe any privacy issues or other relevant ethical considerations.*
 - The dataset does not include personal passenger information, so privacy concerns are minimal.
- *How will the data be represented for the machine learning models? Describe potential needs for data cleaning, feature engineering, scaling, and the like.*
 - Numerical values for times and delays, categorical data for airlines, airport codes, and cancellation reasons. Data cleaning might involve dealing with null values, outliers, and feature engineering like creating a "TotalDelay" column.

MODELING

- *Describe which machine learning models you will explore. Describe how you plan to estimate baseline performance and baseline behavior. Remember that your first models should be simple. Baseline performance can typically be estimated using simple models or even non-machine learning-based solutions. You can also search for results obtained by others on the same or a similar task. You can also estimate “human-level performance” if relevant. Describe how you plan to investigate prediction mistakes and “feature importance” and how this will be used to improve your results.*
 - Linear regression for predicting delay times, logistic regression, and decision trees for classifying flights as ‘delayed’ or ‘on-time’

- Initial estimates will be based on simple statistics from the dataset, like the average delay duration. Human-level performance can be estimated based on dispatchers' manual predictions.
- Examine feature importance to identify the most impactful variables. Investigate incorrect predictions to refine the model.

DEPLOYMENT

- *How will the model(s) be deployed? How will the predictions be used? What are your plans for monitoring and maintaining the machine learning system? If relevant, how do you plan to improve the system after deployment?*
 - There was made an attempt to deploy the model using flask but because of technical difficulties this was not possible.
 - Integrate the model with flight scheduling software to provide real-time delay predictions.
 - Predictions will be used by schedulers and operators to anticipate and manage delays.
 - Set up alerts for when the model's prediction deviates significantly from actual outcomes. Periodic retraining of the model with fresh data.

Showing how the model may be used by OCC:

<https://youtu.be/yNKLjNNhJFU>

REFERENCES

- *List sources you've used during the planning of the project. The list of references should indicate the feasibility of your project.*
 - U.S. Department of Transportation Bureau of Transportation Statistics
 - [Dataset Variable definition](#)
 - [Flight Delay EDA by ADRIÁN VERA](#)
 - Chat GPT, in forms of idea generation, error solving and deploying as I ran into some errors with one of the libraries used.

Based on this, can you give some tips and inspiration to what i can create? I'd like to have something thats aviation or motorsport based



ChatGPT

Certainly! Given your interest in aviation and motorsport, here are some ideas and associated tips for a machine learning project:

1. Aviation-Based Ideas:

a. Flight Delay Prediction:

Description: Develop a model to predict flight delays based on historical data, weather conditions, and other relevant parameters.

Tips:

- **Data Collection:** Use historical flight data which often includes factors like time of day, airline, origin, destination, and previous flight delays.
- **Feature Engineering:** Incorporate external data such as weather forecasts, holidays, and major events.
- **Deployment:** Create a user-friendly app where travelers can input their flight details and get a delay prediction.

b. Aircraft Recognition from Images:

Description: Develop a model that can identify different types of aircraft from images.

Tips:

- **Data Collection:** There are several public datasets containing images of various aircraft.
- **Modeling:** Consider using a convolutional neural network (CNN) for this image classification task.
- **Deployment:** Implement a web application where users can upload an image of an aircraft and get its identification.

2. Motorsport-Based Ideas:

a. Race Outcome Prediction:

Description: Use historical race data to predict the outcome of a motorsport race.

Tips:

Example of how I used for idea generation

ValueError: The least populated class in y has only 1 member, which is too few. The minimum number of groups for any class cannot be less than 2.



ChatGPT

The error you're encountering is due to the stratification process in the `'train_test_split'` method. The error specifically states:

```
python Copy code  
  
ValueError: The least populated class in y has only 1 member, which is too few. The m
```

This happens when you are trying to split the data in a stratified manner (i.e., maintaining the same distribution of target classes in both training and test datasets) and one of your classes in the target column ('ArrDelay' in your case) has only 1 instance. Stratification requires at least 2 instances of every class to be able to split them across training and test datasets.

Here are a few solutions to this issue:

1. **Check Data Distribution:** Before performing the split, check the distribution of your target column. If any class indeed has only one instance, consider removing it, merging it with another class, or obtaining more data for that class.

```
python Copy code  
  
print(df['ArrDelay'].value_counts())
```

2. **Non-stratified Split:** You can split your data without stratification. This will not ensure the same distribution of classes in train and test datasets, but it will avoid the error. To do this, make sure the `'data_split_stratify'` parameter in your `'setup'` function is set to `'None'`.
3. **Manually Handle Stratification:** If you still want to ensure a stratified split, you can manually handle the classes with only one instance. This might involve either duplicating the single instance (though this is not a recommended approach as it introduces artificial data) or combining it with another class.
4. **Use Oversampling or Undersampling:** If the class imbalance is severe and affects multiple classes, consider using techniques like oversampling (adding more instances of the minority class) or undersampling (reducing instances of the majority class) to balance out the classes. There are several methods like SMOTE or ADASYN for oversampling.

Given the context, it's essential to understand the nature and importance of the target classes in your dataset before choosing an approach. If some classes have very few instances, it might be challenging to train a model that predicts those classes accurately.

Example of how I used GPT for errors



ChatGPT

Yes, you can create a Gradio app quickly in Jupyter Notebook using a pre-trained model. Gradio is an excellent tool for creating simple, interactive web interfaces for machine learning models. Here's a basic outline of the steps you'll need to follow:

1. **Install Gradio:** If you haven't already installed Gradio, you can do it using pip. In a Jupyter Notebook cell, run:

```
python
!pip install gradio
```
2. **Import Gradio:** In your notebook, import Gradio:

```
python
import gradio as gr
```
3. **Load Your Model:** Load your pre-trained model. The method for this depends on the type of model and how it's saved.
4. **Create an Interface Function:** Define a function that takes in input(s) and returns the model's output(s). This function will be the interface between your model and the Gradio app.
5. **Create a Gradio Interface:** Use `gr.Interface()` to create the interface. You'll need to pass the function you created, the input types, and the output types. For example:

```
python
interface = gr.Interface(fn=my_model_function, inputs="text", outputs="text")
```
6. **Launch the Interface:** Finally, launch the interface with `interface.launch()`. This will create a web interface in your notebook.
7. **Interact with the Model:** Once the interface is up, you can interact with your model directly from the notebook.

Here's a simple example assuming you have a text-based model:

```
python
# Import Gradio
import gradio as gr

# Suppose you have a function that uses your model
def my_model_function(input_text):
    # Your model processing here
    return processed_output

# Create Gradio Interface
interface = gr.Interface(fn=my_model_function, inputs="text", outputs="text")

# Launch the Interface
interface.launch()
```

Remember, the exact implementation will depend on the specifics of your model and the type of inputs and outputs it requires.

Example of how I used GPT to deploy the model.