Flight Quest: Flight Optimization, Paper 36

Alexander Guziel (asg252), Atheendra PT (ap778), Rene Zhang (rz99), Bo Yuan Zhou (bz88)

Abstract

1 Introduction

In the modern day, airplanes are a frequently-used means of transportation. However, using it is fairly expensive to manage due to costs of fuel and delay. Due to many different factors, such as weather, restricted areas, and jet streams, it's very difficult to keep this cost of such transportation at an optimal level; airline companies are always looking for more ways to optimize their flights. What we aim to do here in Flight Quest 2 is to optimize to the best of our ability the costs of airplane flights using machine learning techniques on large sets of data. Flight Quest 2 is a competition on Kaggle hosted by General Electric. With thin margins, a small reduction in cost can cause large increases in profit. Reducing cost can also make flying easier for people and can also help the environment by reducing fuel burn. The goal in this competition is to create an agent that generates flight routes based on certain information given. The problem formulated is that the training data gives parameters such as flight routes and broadcasts from the airplane. In real life, we have all this information, up until a certain point in time. This is how the test data is structured. (perhaps we should move this) The goal is to create an agent that can advise pilots on optimal flight patterns. (let's add some sources about costs) (I think this can be combined with motivation)

1.1 Motivation

Our motivation for undertaking this task came from searching for something that would challenge us to apply what we learned in innovative ways, give us experience with working on a real-life machine learning problem, and give us an opportunity to have an impact on people's lives. The Flight Quest problem satiates these desires by starting us off at the beginning, giving us a goal, and then telling us to go about solving this problem however we deem necessary. It presents to us a problem that we have never encountered; a problem that requires learning to create optimization instead of classification. Therefore, our attempts at solving this problem are more centered around finding ways to extend and evolve the machine learning techniques that we've been using for other learning tasks rather than simply reuse them. This project requires more creativity as the literature on this problem is very limited. We also deal with the real-life issues of feature selection and generation.

2 Problem Definition and Methods

2.1 Task

Our task finding least-cost paths for new flights to take from their current location to get to their location. We are given that the following things factor into calculating the cost of a flight:

- Fuel consumption
- Delay of flight arrival
- Crossing into restricted areas
- Changing the speed and altitude many times

However, we are not given how much the factor in, and there are still more factors involved in the cost that aren't as clear-cut, such as turbulence and weather effects.

In order to assist us in finding an optimal path, we have data from previous flights and other things such as airport and restricted area locations. However, the size of the data is very large (several gigabytes worth), meaning that we will have to be selective with what we use. Beyond this, there are no other guidelines to follow; we are free to approach the problem in any way necessary and use any resources that we need for our approach.
This problem is important because in this age big data can be seemingly used to optimize a processes so it is worth asking whether it can help air-

lines. This problem is interesting because we wonder whether an agent can indeed help pilots know which routes to follow.

The basic idea is given the flight with the following parameters (add later), that we construct a list of 'waypoints' which are points the aircraft flies to consisting of latitude, longitude, and altitude. We also specify what speed the aircraft should travel at at that point. (let's list all the data we have here)

The competition is scored by submission onto a website. On their end, they run a simulator with weather data and ground conditions data which uses a physical model to simulate fuel burn using factors such as windspeed and distance traveled. Because of the nature of this, techniques such as cross validation tend to not work and overfitting is less of a concern because of the large hypothesis space. It is also difficult to make generalizations about the generalization error because our "error" in this case is deviation from the optimal cost, which is unknown.

2.2 Algorithms and Methods

The basic goal in this case should be to create a series of points and classify them with altitude and speed for each aircraft. Our principal assumption is that the great-circle path should be the shortest, provided we avoid restricted zones which incur the same cost as the flight not reaching the destination. We see k-nearest neighbors as a very natural solution as well as ensemble methods due to their robustness.

2.2.1 Restricted Zone Avoidance

There were many reasons why our baseline did not produce the most optimal path, but the largest aspect we overlooked in our previous assumptions was the existence of restricted areas, which are locations on maps that passenger planes are not allowed to pass through i.e. passing through them would results in accumulating large amounts of cost. Therefore, we want to maintain the shortest path as we did with the baseline, but this time take into consideration the location of restricted areas.

Another way to avoid restricted zones is to follow our own path. In order to do this, we took the following approach: we generate using a uniform distribution many "waypoints" that lie between the current and detination location geographically. Then, we utilize an A* search to find the shortest path from our current location to our destination location and avoid restricted zones by employing a distance function that sets the distance from any point to a point in a restricted zone to be very large.

2.2.2 Condition Optimization

The distance that the path covers is only one factor for the cost; in order to have a complete flight plan, we must also include speeds and altitudes for each part of the flight plan. Both of these also affect the cost of our flight in many ways. For example, if the flight is on track to arrive on time, we don't want to unnecessarily expend fuel to get there faster. We also want our flight plans to be as consistent with these attributes as possible, since changing them multiple times incurs cost during the transitions. Keeping this in mind, we made several varying attempts in our approach. (I'm not sure this belongs here)

2.2.3 Data Trimming and Feature Selection

Before we began using any machine learning models, we had to determine how we would first handle the data. As stated before, the data provided to us is very large and contains some unimportant information, so training on a model using all of it is time-expensive. We first identified the features that were given by the test flights, which were substantially fewer than those given by the training flights and then took those into consideration. Afterwards, from these trimmed down features, we used those that would best match the method we used.

2.2.4 Whole Flight Classification

For many flights, we made the assumption that for a certain distance of flight, there is an optimal speed and altitude to cruise at at which the airplane will follow for the majority of the route which excludes descent and ascent.

2.2.5 kNN Modeling

If we assume that our flight examples finished with an optimal path, then one way to treat this problem is, for each test flight, to find the most similar flight path from history and follow what that flight did. In order to do this, we would need to first define a similarity measure for flights. Once we do that, we choose the first nearest neighbor flight path and follow that path. We did not consider using a larger value of k due to the fact that taking the weighted average for flight paths would be suboptimal, as an average of paths could take us into a restricted area, among other things.

2.3 Point by Point Classification

2.3.1 Ensemble Methods

2.3.2 kNN Modeling

Similar to how we approached the restricted zones problem, we can model our test flight by comparing it to a previous training flight.

3 Evaluation

3.1 Methodology

Since the goal is to create a path of least cost, our approach remains consistent in that we prioritize optimizing things that will have the biggest impact on the cost. However, since we don't know for sure all of the factors that go into calculating the cost nor how much each one is considered, we must also reorganize our plan based on assumptions made *à la* a trial-and-error process.

3.1.1 Baseline

We used the simplest assumption for our baseline that since fuel was one of the considerations for calculating cost, so if we optimize the distance traveled (i.e. find the shortest-distance path), then we conserve fuel and as a result, lower our cost. However, since we are working with spherical coordinates instead of Euclidean coordinates, we must consider a different way of determining the shortest distance than what we are used to. For spherical coordinates, this distance is called the "great-circle" distance and is found by [3]:

1. Finding the great circle on which the two points (current position and destination) lie.
2. Separating the great circle into two arcs via the two points.
3. Taking the smaller of the two arcs.

Despite our assumptions, using this method to determine our answer incurred a cost that was significantly large (see table in section 3).

3.2 Results

Due to the fact that our problem was one of optimization, the only way for us to analyze how "correct" our solution was was to utilize a cost calculator, which would take in a set of completed test flights and return a score based on the total cost incurred across all of the test flights. The full results can be found in figure 1.

3.3 Discussion

4 Related Work

We were not able to find any academic literature on this. We were able to peruse the forums for this competition and milestone version of it. Top-ranked submissions for the milestone used the data to train for optimal cruise speed and altitude. They also trained for optimal descent. Some used a cost-based model. The milestone competition winner used a cost based model which considered only the features that could reduce fuel burn.

5 Future Work

While we have done work and seen improvements, a lot can be done. It is likely that optimal routes may not just the shortest geographic distance due to winds. But the winds of the future cannot be known ahead of time. Being able to forecast wind and relating a cost function to wind over the entire geographic map may allow an improvement in path. One way may to be use KNN to see what direction a plane at the current point will head. This is a bit problematic though as bearing changes along a great-circle path from one point to another. Initial bearing may be used or perhaps the classification should be what waypoint should the plane head to next. It may also help to consider other methods in classification of speed and altitude. Currently, fuel cost and delay cost is not present in our model. These features, if added, could definitely yield improvements but the question of how to implement them is varied since there is not a direct cost to fuel cost relation and the cost of delays and fuel is not included in the training data. The optimal way would likely be to incorporate this into our model as parameters that can be tweaked with weights in a higher level learning problem to determine the optimal weights.

Another issue is the presence of outliers in our data classification. Inspecting our paths, most paths seem very reasonable but sometimes outliers are present in the data where a plane goes from 36000 feet altitude to 8000 feet to 36000 feet in two steps. Detecting changes like this aren't quite that simple. It may be possible to train these using

some type of Markov model where the probability of different percentage based or absolute altitude changes are calculated and we tweak the confidence that this is a mistake to smooth the data. Then, the middle point could be the arithmetic or geometric mean of the two points' altitudes.

6 Conclusion

7 References

References

[1] "Description-Flight Quest 2: Flight Optimization — GE Quest". http://www.gequest.com/c/flight2-main.

[2] "Flight Quest 2: Flight Optimization Forum — GE Quest". https://www.gequest.com/c/flight2-milestone/forums.

[3] "Great-circle Distance". http://en.wikipedia.org/wiki/Great-circle_distance.

[4] "scikit-learn: machine learning in Python". http://scikit-learn.org/stable/.