Machine Learning Assignment 54

Elijah Tarr

February 24, 2021

Problem 1

(1a) **Part a is in the code, but these are the distances**

```
[{'type': 'Shortbread', 'dist': 0.04690415759823429},
{'type': 'Shortbread', 'dist': 0.037416573867739396},
{'type': 'Shortbread', 'dist': 0.061644140029689765},
{'type': 'Shortbread', 'dist': 0.1224744871391589},
{'type': 'Sugar', 'dist': 0.15811388300841897},
{'type': 'Sugar', 'dist': 0.158113883008419},
{'type': 'Sugar', 'dist': 0.08831760866327845},
{'type': 'Sugar', 'dist': 0.24494897427831783},
{'type': 'Fortune', 'dist': 0.21213203435596428},
{'type': 'Fortune', 'dist': 0.18708286933869708},
{'type': 'Fortune', 'dist': 0.3959797974644666},
{'type': 'Fortune', 'dist': 0.17320508075688776},
{'type': 'Fortune', 'dist': 0.22759613353482086}]
```

(1b) **This is the code output for part b**

```
[{'type': 'Shortbread', 'dist': 0.037416573867739396, 'id': 2},
{'type': 'Shortbread', 'dist': 0.04690415759823429, 'id': 1},
{'type': 'Shortbread', 'dist': 0.061644140029689765, 'id': 3},
{'type': 'Sugar', 'dist': 0.08831760866327845, 'id': 7},
{'type': 'Shortbread', 'dist': 0.1224744871391589, 'id': 4}]
```

So the 5 nearest cookies are 4 short breads with ID's [1, 2, 3, 4] and 1 sugar cookie with ID 7.

(1c) **Which type of cookie showed up most?**

The type of cookie that was nearest to this point was the short bread cookie, which makes me think that this recipe will actually produce a short bread cookie!

(2a) **What happens if the data set is too large?**

If the k amount of nearest neighbors is too large, the sheer amount of each type of cookie in our data set could skew the result. If the nearest neighbors amount includes the entire dataset, then no matter where the point is that we want to test, it will always give back the same result, since the nearest neighbors are always the same. This makes the test useless.

(2b) **For each classification of cookie, find the average distance between P and the points corresponding to the cookies in that classification. Explain how this resolves the issue you identified in part (a).**

When we get the average distance from the point to each type of cookie, we eliminate the need to search through the list to find the nearest neighbors, so now all we need to do is find the one with the minimum distance. The only problem that arises, however, is if the average distance to one group for a certain point equals the average distance to another group, where you would have to choose a group.

Problem 2

(a) **Create a quantitative data set to represent this information, and include it in your write up. Name your features appropriately.**

```
[[1, 95, 33, 1],
[0, 95, 34, 0],
[1, 92, 35, 1],
[0, 85, 30, 1],
[1, 80, 36, 1],
[0, 85, 29, 1],
[1, 95, 36, 1],
[0, 87, 31, 1],
[1, 99, 36, 0],
[0, 95, 32, 0]]
```

This python list is the quantitative data set I will use to represent my information. The first column is whether the person was accepted into the college or not, 1 if so and 0 otherwise. The second column is the percentile of their class they were in. The third column is their ACT score, and the fourth column is slightly subjective, but objective from the data I was given. If they had done some extra-curricular thing like sports, internships, business, or some subject-olympiad-thing, they got a 1. Otherwise, a 0. Each row is each person.

The columns will be named: [Accepted, Percentile, Score, Academic EC, Sports EC]. The prediction column will be the accepted column, because that is what we will be trying to predict in this problem.

(b) **Decide what type of model you will use to model the probability of acceptance as a function of the features in your dataset. State and justify the form of the model in your writeup.**

The prediction column in our data is either a 0 or a 1. It could represent the probability of acceptance, because everyone who got accepted has a 100% chance of being accepted. The best model for a column that must be in [0, 1] is definitely a logistic regression. This is because the outputs are constrained to just that: [0, 1]!

(c) **Fit the model to the data. For each feature, answer the following questions: According to your model, as that variable increases, does the estimated probability of acceptance increase or decrease? Does that result make sense? If so, why?**

The coefficients are 'percentile': -0.0666803, 'score': -0.4560402, 'academicec': -1.7243942, 'sportsec': -1.6684874, 'const': 22.3909992.

The prediction equation is $\frac{1}{1+e^{(\sum_n \beta_n * p_n)}}$, and we can find a couple patterns here. If the exponent of e is negative, then the denominator will be closer to 1, making the outcome closer to 100%. Therefore, the more negative the coefficient, the more it matters to if the person will get into college or not.

This means that you will have a higher chance to get in if you are in a top percentile of your class and get a good score on the ACT, but the most important factor is doing an academic or sports extra curricular activity.

This pattern can be reversed, so the more positive the exponent, the larger the denominator will be, and the smaller the outcome will be. The constant coefficient is very highly positive, so that if you get a 0 on the ACT, you are in the 0th percentile of your class, and you do 0 extracurriculars, the constant coefficient will make you have a very low percentage of getting in.

(d) **Estimate the probability of being accepted for each of the data points that you used to fit the model. How well does this match up with reality?**

Estimating the probabilities of being accepted for each college is fairly accurate. If the person got accepted, we will get a higher (60%+) chance of getting in. However, if they didn't get accepted, they would get a lower (¡40%) chance of getting in. It turns out that there is one key factor for how accurate this percentage will be, and that is the correction. The logistic regressor will have a domain error if an input of 0 or 1 exactly is put through it, so our solution is to change it to be very small, such as 0.00001 instead of 0, and 0.99999 instead of 1. This way, the model can actually train. But, the difference between 0.01 and 0.000000001 is very large, it can make a 99% into close to a 60%, so we have to be careful of how small we set that value.

(e) **Estimate your probability of being accepted if you are in the 95th percentile of your class and got a 34 on the ACT. Justify why your model's prediction is reasonable.**

The probability of being accepted into the college if you are in the 95th percentile and got a 34 on the ACT is around 37%. This is pretty reasonable. The majority of our datapoints had a percentile of 95 or higher, but the only person who got in without any extracurriculars did exceptional on the ACT and was in the 99th percentile. Therefore it is pretty unlikely because you aren't doing any extracurriculars.

(f) **Now suppose that you have an opportunity to do an internship at a well-known company the summer before you apply to college. If you do it, what will your estimated probability of acceptance become? Based on this information, how much does the internship matter in terms of getting into the college you want?**

If you take the internship, your estimated probability of acceptance becomes close to 76%. This is a very drastic change (probably due to the ”small constant” I talked about earlier) but it almost guarentees you to get in.

Based on this information, the internship has a great impact on whether you get in or not.