Ben Tankus

CS 519 HW 1

4-2-21

**1 Data (Pre-)Processing (Feature Map)**

1. **Take a look at the data. A training example looks like this:**

**37, Private, Bachelors, Separated, Other-service, White, Male, 70, England, <=50K**

**which includes the following 9 input fields plus one output field (y): age, sector, education, marital-status, occupation, race, gender, hours-per-week, country-of-origin, target Q: What are the positive % of training data? What about the dev set? Does it make sense given your knowledge of the average per capita income in the US?**

Train Positive Percent is: 1251 of 5000 or 25%

Dev Positive Percent is: 236 of 1000 or 23.6%

These values make sense for yearly salaries in the 90s. According to google, median salary in the 90s was around 20,000 dollars so 25% of people around $50,000 makes sense

1. **Q: What are the youngest and oldest ages in the training set? What are the least and most amounts of hours per week do people in this set work? Hint: cat income.train.txt.5k | sort -nk1 | head -1**

Min Hours 1 min age 17

Max Hours 99 max age 90

**3. There are two types of fields, numerical (age and hours-per-week), and categorical (everything else). The default preprocessing method is to binarize all categorical fields, e.g., race becomes many binary features such as race=White, race=Asian-Pac-Islander, etc. These resulting features are all binary, meaning 1 In principle, we could also convert education to a numerical feature, but we choose not to do it to keep it simple. 1 their values can only be 0 or 1, and for each example, in each field, there is one and only one positive feature (this is the so-called “one-hot” representation, widely used in ML and NLP). Q: Why do we need to binarize all categorical fields?**

Answer: The simple reason we need to binarize all categorical fields is for math. One cannot sum, average, or do math of much significance on categorical data without transforming it to numerical (binarizing).

**4. Q: If we do not count age and hours, what’s maximum possible Euclidean and Manhattan distances between two training examples? Explain.**

Answer: Euclidean distance: sqrt(2x7) = **3.7417**, Manhattan: 2x7 = **14**

There are 7 total fields not including age and hours. When binarized, each of these fields have two options, true or false. Assuming the two furthest apart points, their difference will be a function of number of fields, and number of choices, in this case, 2x7. Manhattan distance is just this simple product, Euclidean distance is the square root of this.

**5. Why we do not want to binarize the two numerical fields, age and hours? What if we did? How should we define the distances on these two dimensions so that each field has equal weight? (In other words, the distance induced by each field should be bounded by 2 (N.B.: not 1! why?)).**

**Hint: first, observe that the max distance between two people on a categorial field is 2. If we simply “normalize” a numerical field by, say, age / 100, it might look OK but now what’s the max distance between two people on age? Are you treating all fields equally?**

Answer: The two numerical fields already can be analyzed without binarizing. The range of both these fields is great enough that binarizing them would create excessive model complexity, greatly increasing the terms in the model. For instance, there are ages ranging from 17 to 90 in the data, meaning there would be roughly 70 additional columns if this field were binarized, for little added value. The same is true for hours. To ensure equal weight to both fields I recommend transforming them both to be percent of maximum fields, where the maximum column value is considered 100, and each data point is a percent of that.

**6. Q: How many features do you have in total (i.e., the dimensionality)?**

**Hint: should be around 90. How many features do you allocate for each of the 9 fields?**

**Hint: for i in `seq 1 9`; do cat income.train.txt.5k | cut -f $i -d ',' | sort | uniq | wc -l; done**

Answer: In the Dev set there are 88 features not including the numerical features. Including the numerical features there should be around 235. This changes to 92 and 230 respectively in the training set. These numbers are different because of the different categories available in each.

**7. Q: How many features would you have in total if you binarize all fields?**

Answer: If all fields were binarized there would be 235. If only the numerical features were binarized there would only be 88. In the training set it would be 230 and 88 respectively.

**PART 2**

**1. Q: Find the five (5) people closest to the last person (in Euclidean distance) in dev, and report their distances:**

Original Person:

['58', 'Private', 'HS-grad', 'Widowed', 'Adm-clerical', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 3772 e-distance: 2.646

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Adm-clerical', 'Black', 'Female', '40', 'United-States', '<=50K']

Index: 999 e-distance: 2.567

person: ['38', 'Private', 'Bachelors', 'Married-civ-spouse', 'Sales', 'White', 'Male', '42', 'United-States', '>50K']

Index: 4931 e-distance: 2.646

person: ['21', 'Private', 'Assoc-voc', 'Never-married', 'Transport-moving', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 3855 e-distance: 2.646

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Exec-managerial', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 31 e-distance: 2.646

person: ['21', 'Private', 'Some-college', 'Never-married', 'Machine-op-inspct', 'White', 'Male', '40', 'United-States', '<=50K']

I’m pretty sure this is wrong because it doesn’t quite match the original. Especially in age. It’s very close in hours worked and the categorical values though.

**2. Redo the above using Manhattan distance.**

Index: 4999 manhat-distance: 7.0

person: ['21', 'Private', 'Assoc-voc', 'Never-married', 'Adm-clerical', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 4982 manhat-distance: 7.0

person: ['21', 'Private', 'Some-college', 'Never-married', 'Other-service', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 4649 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Adm-clerical', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 718 manhat-distance: 7.0

person: ['21', 'Private', '10th', 'Never-married', 'Handlers-cleaners', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 4170 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Craft-repair', 'White', 'Male', '40', 'United-States', '<=50K']

Again these values seem close to the original, but not exact. Also the distances seem much higher than those we got in the examples. I’ve been to office hours many times and we can’t seem to figure out why my code is producing the wrong numbers.

**3. What are the 5-NN predictions for this person (Euclidean and Manhattan)? Are these predictions correct?**

Euclidean: <= 50K. Correct!

Manhattan: <= 50K. Correct!

**4. Redo all the above using 9-NN (i.e., find top-9 people closest to this person first).**

Original Person:

['58', 'Private', 'HS-grad', 'Widowed', 'Adm-clerical', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 31 e-distance: 2.646

person: ['21', 'Private', 'Some-college', 'Never-married', 'Machine-op-inspct', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 533 e-distance: 2.646

person: ['21', 'Private', 'Some-college', 'Never-married', 'Handlers-cleaners', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 1530 e-distance: 2.646

person: ['21', 'Private', 'HS-grad', 'Married-civ-spouse', 'Handlers-cleaners', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 4999 e-distance: 2.646

person: ['21', 'Private', 'Assoc-voc', 'Never-married', 'Adm-clerical', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 999 e-distance: 2.567

person: ['38', 'Private', 'Bachelors', 'Married-civ-spouse', 'Sales', 'White', 'Male', '42', 'United-States', '>50K']

Index: 3772 e-distance: 2.646

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Adm-clerical', 'Black', 'Female', '40', 'United-States', '<=50K']

Index: 3855 e-distance: 2.646

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Exec-managerial', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 4931 e-distance: 2.646

person: ['21', 'Private', 'Assoc-voc', 'Never-married', 'Transport-moving', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 4575 e-distance: 2.646

person: ['21', 'Local-gov', 'Some-college', 'Never-married', 'Adm-clerical', 'White', 'Male', '40', 'United-States', '<=50K']

**Manhattan**

Index: 4999 manhat-distance: 7.0

person: ['21', 'Private', 'Assoc-voc', 'Never-married', 'Adm-clerical', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 4982 manhat-distance: 7.0

person: ['21', 'Private', 'Some-college', 'Never-married', 'Other-service', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 4649 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Adm-clerical', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 718 manhat-distance: 7.0

person: ['21', 'Private', '10th', 'Never-married', 'Handlers-cleaners', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 4170 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Craft-repair', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 3163 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Transport-moving', 'Black', 'Male', '40', 'United-States', '<=50K']

Index: 2391 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Prof-specialty', 'White', 'Female', '40', 'United-States', '<=50K']

Index: 31 manhat-distance: 7.0

person: ['21', 'Private', 'Some-college', 'Never-married', 'Machine-op-inspct', 'White', 'Male', '40', 'United-States', '<=50K']

Index: 146 manhat-distance: 7.0

person: ['21', 'Private', 'HS-grad', 'Never-married', 'Craft-repair', 'Black', 'Male', '40', 'United-States', '<=50K']

Both methods yielded <=50K for the prediction which is correct.

**PART 3**

**1. Implement the basic k-NN classifier (with the default Euclidean distance)**

Q: Is there any work in training after finishing the feature map

There is no training the k-NN model, so there is no work in training the after finishing the feature map

Q: What’s the time complexity of k-NN to test one example (dimensionality d, size of training set |D|)?

Time complexity steps:

1. Calculating the distance between each person: O(D) \* d (all people)

2. Sort the algorithm: O(D^2)

3. Time to check the top matches: k

4. Overall time complexity: O(D) \* d + D^2 + k

Q: Do you really need to sort the distances first and then choose the top k? Hint: there is a faster way to choose top k without sorting

It would be faster to search the list of distances for minimum and remove them without replacement. This time complexity would simply be D\*k. I’m not really sure how to implement this though.

**2. Q: Why the k in k-NN has to be an odd number?**

K is typically an odd number because it is then less likely for ties to occur when comparing data.

1. **Evaluate k-NN on the dev set and report the error rate and predicted positive rate for k = 1, 3, 5, 7, 9, 99, 999, 9999, e.g., something like:**

**k=1 dev\_err xx.x% (+:xx.x%) k=3 ... ... k=9999 ...**

Q: what’s your best error rate on dev, and where did you get it? (Hint: 1-NN dev error should be ∼23% and its positive % should be ∼27%).

I wasn't able to speed up my algorithm enough to calculate error rate, but it is likely that the best error rate occurs between k = 5 and k = 100. K = 1 is obviously underfitting, where k over 100 is fitting each point in dev using 10% of the data, which seems large, and likely to overfit the model.

1. **Now report both training and testing dev errors:**

???

Q: When k = 1, is training error 0%? Why or why not? Look at the training data to confirm your answer.

When K = 1 the training error would be very close to 0%, usually around 1-2%. This is because we are using only one record to predict the salary, and it’s much less likely we will get the correct value. It is not 0% however, because we are still using one value to predict.

**5. Q: What trends (train and dev error rates and positive ratios, and running speed) do you observe with increasing k? Do they relate to underfitting and overfitting?**

There is certainly a curve to the model accuracy. As K increases it will certainly take the model more time to run. Overfitting will definitely occur at k = n, or 5,000 k’s, where positive rate is 0% and training error is 25% (population average). This happens because there are more negative than positive responses, and when the k’s are larger than the number of records in the data, the algorithm will use the entire data set to predict salary and gets the same value each time (non-positive).

As the value of K increases from 1 to infinity, the error rate will decline to the lowest point, likely between k = 5 and k = 100, then rise again as the model progresses to overfitting.

Q: What does k = ∞ actually do? Is it extreme overfitting or underfitting? What about k = 1?

K = inf is extreme overfitting and will have error = 0 and k-nn prediction = average of the whole sample. k = 1 is extreme underfitting and will have high error and a wildly inaccurate prediction.

6. Redo the evaluation using Manhattan distance. Better or worse? Any advantage of Manhattan distance?

??

7. Redo the evaluation using all-binarized features (with Euclidean). Better or worse? Does it make sense?

??

**PART 4**

Now try more k’s and take your best model and run it on the semi-blind test data, and produce income.test.predicted, which has the same format as the training and dev files.

Q: At which k and with which distance did you achieve the best dev results?

Q: What’s your best dev error rates and the corresponding positive ratios?

Q: What’s the positive ratio on test?

Part of your grade will depend on the accuracy of income.test.predicted.

**PART 5**

5 Observations

**1. Q: Summarize the major drawbacks of k-NN that you observed by doing this HW. There are a lot!**

The major k-NN drawbacks are that it is a very computationally expensive algorithm that doesn’t provide the best results. It’s fairly trivial to understand, and one of the only machine learning methods that you can really visualize what it’s doing well, but it’s computationally difficult to implement, and can’t be used well for predictions. K-NN also requires the storage of the whole dataset in memory, which makes it impossible to use for large data sets.

**2. Q: Do you observe in this HW that best-performing models tend to exaggerate the existing bias in the training data? Is it due to overfitting or underfitting? Is this a potentially social issue?**

**???**

**3. Q: What numpy tricks did you use to speed up your program so that it can be fast enough to print the training error? Hint: (a) broadcasting (such as matrix - vector); (b) np.linalg.norm(..., axis=1); (c) np.argsort() or np.argpartition(); (d) slicing. The main idea is to do as much computation in the vector-matrix format as possible (i.e., the Matlab philosophy), and as little in Python as possible.**

1. **How many seconds does it take to print the training and dev errors for k = 99 on ENGR servers? Hint: use time python ... and report the user time instead of the real time. (Mine was about 14 seconds).**
2. **What is a Voronoi diagram (shown in k-NN slides)? How does it relate to k-NN?**

A Voronoi diagram is one of the best ways to visualize the k-NN method. You can think of it as a mosaic, where each mosaic piece (Voronoi Cell) is the region defined by a nearest-neighbor outcome.

**Debriefing (required in your report)**

1. Approximately how many hours did you spend on this assignment?

A lot. I time myself when working on classes and I spent X time on this assignment in the first three weeks of the class.

1. Would you rate it as easy, moderate, or difficult?

Immensely difficult. I consider myself pretty good at code for a data person. Professionally I am currently a Data Analyst for a LinkedIn company and code for them every day. I fairly frequently implement AI algorithms in my work, using the typical AI packages. In my academic life I performed very well in both CS 511 and 512, getting 101% and 97.5% in each class respectively.

I understand the need to prevent the use of pandas/scikitlearn in this assignment. It certainly taught me a lot, but most of it was through other students and going to office hours. There were not enough resources available to aid me in this assignment. Many of the questions I didn’t even understand what it was asking, and it was VERY different from lecture. There was a lecture where we were walked through binarization, but it just didn’t work on my computer, and nothing past part 3 was discussed anywhere in the class.

Honestly I feel like I’m missing a massive section of content, and this whole class relies on some knowledge that I just don’t have.

1. Did you work on it mostly alone, or mostly with other people?

Mostly alone with a bit of help from the TA’s and other students.

1. How deeply do you feel you understand the material it covers (0%–100%)?

I understand K-NN and Euclidean distance 100%, and I understand how to call pd.getdummies 100%, probably 15% better than I did before this homework.

I absolutely have no idea how binarization in numpy works, and how/what the mapping function does.

1. Any other comments?

* Start the class with the windows guide. I lost about a week because I could not follow lecture with my windows computer.
* Ask questions that you teach us how to answer. I felt completely unprepared on most of these questions. Most of the lecture is “here is numpy” and the homework is expecting a high level of numpy knowledge.
* I appreciate the TA’s willingness to help, the slack channel has been very helpful.