

III. Model documentation and write-up

Who are you (mini-bio) and what do you do professionally?

Algotrader.

What motivated you to compete in this challenge?

I developed my version of decision trees and wanted to check it in a big competition.

High level summary of your approach: what did you do and why?

Various ngram (k-mer in biology) kernels, naive Bayes, soft masks, and rank model's merging. I mostly go for fast solutions. I got to top7 on my first day with some one-hour ngram solution. Later I tweaked kernel to be +4% better and time down to 4 mins. Actually, I can get to top5 in 5 mins, but there are no prizes.

Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

This the ngram kernel (T3Model.cs), all genetic processing relies upon it:

```
int iStart = 0;
int jStart = 0;

for (int i = 0; i < testPolymers.Length; i++) {
    if ((i == testPolymers.Length - 1)
        || (testPolymers[i] != testPolymers[i + 1])) {
        while ((jStart < trainPolymers.Length)
            && (trainPolymers[jStart] < testPolymers[i])) jStart++;

        if (jStart == trainPolymers.Length) { break; }

        if (trainPolymers[jStart] != testPolymers[i]) {
            iStart = i + 1;
            continue;
        }

        int jEnd = jStart;

        while ((jEnd < trainPolymers.Length)
            && (trainPolymers[jEnd] == testPolymers[i])) jEnd++;
```

```

for (int ik = iStart; ik <= i; ik++) {
    bool bNewPolymer = true;

    for (int l = 0; bNewPolymer && (l < devSetTops.Length); l++) {
        for (int j = jStart; j < jEnd; j++) {
            if (trainLabs[j] == devSetTops[l][testPlasmids[ik]]) {
                bNewPolymer = false;
                break;
            }
        }
    }

    if (bNewPolymer) {
        for (int j = jStart; j < jEnd; j++) // ADD WEIGHTS !!!
            resLabs[testPlasmids[ik]][trainLabs[j]] += weights[j];
    }
}

iStart = i + 1;
jStart = jEnd;
}
}

```

This is the soft masks kernel (SoftMasks.cs), best of my non-genetic models:

```

for (int i = 0; i < resLabs.Length; i++) {
    ulong bits = testColumnsULong[i];

    if (!forceLabsDictionary.ContainsKey(bits)) {
        List<KeyValuePair<int, int>> labsList =
            new List<KeyValuePair<int, int>>();

        for (int j = 2; j < 20; j++) {
            foreach (var keyValue in forceLabsDictionary) {
                ulong newBits = keyValue.Key;

                newBits = newBits ^ bits;

                if (Util.PopCount(newBits) < j) labsList.AddRange(keyValue.Value);
            }

            if (labsList.Count > 0) { break; }
        }
    }
}

```

```

    }

    forceLabsDictionary[bits] =
        labsList.OrderByDescending(v => v.Value).ToArray();
}

var labs = forceLabsDictionary[bits];
double alpha = Math.Pow(labs.Length, pow);

for (int j = 0; j < labs.Length; j++) // UP RANKS !!!
    resLabs[i][labs[j].Key] += 1d / (alpha + j);
}

```

This is a merging of two models (Merges.cs/RankSum2):

```

for (int i = 0; i < src1.Length; i++) {
    int[] orders1 = src1[i].MyOrderDescending();
    int[] orders2 = src2[i].MyOrderDescending();

    resLabs[i] = new double[src1[i].Length];

    for (int j = 0; j < orders1.Length; j++) { // SUM RANKS !!!
        resLabs[i][orders1[j]] += alpha / (j + 1f);
        resLabs[i][orders2[j]] += (1 - alpha) / (j + 1f);
    }
}

```

Please provide the machine specs and time you used to run your model.

CPU (model): Intel i7-3770K

GPU (model or N/A): N/A

Memory (GB): 16GB

OS: Windows 7

Train duration: N/A

Inference duration: 40 minutes

What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

Tried my decision trees library for merging. It still works better than XGBoost, but simple rank merges work even better. Tried advanced ngram weight formulas.

Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

My C# research library. It is not complicated but fits my needs. It turns C# into something like Python in terms of research speed. Fewer libraries, but more speed for a custom code. Fun fact: I'm probably the first and only C# user prize winner of big ML competitions.

How did you evaluate performance of the model other than the provided metric, if at all?

Not really.

Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

It's RAM hungry. It creates 20+Gb on SSD, and you have to use SSD. And you have to clean up caches if you change the test file.

Do you have any useful charts, graphs, or visualizations from the process?

Tons of incomprehensible figures and texts, but not charts.

If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

I definitely would try neural nets for non-genetic and merging. I would not add more stacking, because it's hard to use stacking in production.