chimeras

Chimera detection

Datasets: UNITE, Silva?

Silva nemá ITS asi - ale mžeme použít to eukaryome databasene

List of all UCHIME reference datasets

Version no	Release date	No of sequences	Release status	Link	Notes
9.0	2022-10-16	161 335	Current	**https://doi.org/10.15156/BIO/2483933**	**When using this resource, please cite it as follows:** Abarenkov, Kessy; Zirk, Allan; Piirmann, Timo; Pöhönen, Raivo; Ivanov, Filipp; Nilsson, R. Henrik; Kõljalg, Urmas (2022): UNITE UCHIME reference dataset. Version 16.10.2022. UNITE Community. **https://doi.org/10.15156/BIO/2483933**
7.2	2017-06-28	30 555		**Download**	
7.1	2016-12-01	29 342		**Download**	
7.0	2016-01-01	22 774		**Download**	
7.0	2015-03-11	22 219		**Download**	
6.0	2014-07-26	21 059		**Download**	

List of all USEARCH/UTAX/SINTAX reference datasets

Version no	Release date	Taxon group	No of sequences	Release status	Link	Notes
10.0	2025-02-19	Fungi	168 030	Current	**https://doi.org/10.15156/BIO/3301245**	**When using this resource, please cite it as follows:** Abarenkov, Kessy; Zirk, Allan; Piirmann, Timo; Pöhönen, Raivo; Ivanov, Filipp; Nilsson, R. Henrik; Kõljalg, Urmas (2025): UNITE USEARCH/UTAX release for Fungi. Version 19.02.2025. UNITE Community. **https://doi.org/10.15156/BIO/3301245**
10.0	2025-02-19	All eukaryotes	266 589	Current	**https://doi.org/10.15156/BIO/3301246**	**When using this resource, please cite it as follows:** Abarenkov, Kessy; Zirk, Allan; Piirmann, Timo; Pöhönen, Raivo; Ivanov, Filipp; Nilsson, R. Henrik; Kõljalg, Urmas (2025): UNITE USEARCH/UTAX release for eukaryotes. Version 19.02.2025. UNITE Community. **https://doi.org/10.15156/BIO/3301246**
10.0	2024-04-04	Fungi	159 195		**https://doi.org/10.15156/BIO/2959340**	**When using this resource, please cite it as follows:** Abarenkov, Kessy; Zirk, Allan; Piirmann, Timo; Pöhönen, Raivo; Ivanov, Filipp; Nilsson, R. Henrik; Kõljalg, Urmas (2024): UNITE USEARCH/UTAX release for Fungi. Version 04.04.2024. UNITE Community. **https://doi.org/10.15156/BIO/2959340**
10.0	2024-04-04	All eukaryotes	252 239		**https://doi.org/10.15156/BIO/2959341**	**When using this resource, please cite it as follows:** Abarenkov, Kessy; Zirk, Allan; Piirmann, Timo; Pöhönen, Raivo; Ivanov, Filipp; Nilsson, R. Henrik; Kõljalg, Urmas (2024): UNITE USEARCH/UTAX release for eukaryotes. Version 04.04.2024. UNITE Community. **https://doi.org/10.15156/BIO/2959341**
9.0	2023-07-18	Fungi	206 494		**https://doi.org/10.15156/BIO/2938083**	**When using this resource, please cite it as follows:** Abarenkov, Kessy; Zirk, Allan; Piirmann, Timo; Pöhönen, Raivo; Ivanov, Filipp; Nilsson, R. Henrik; Kõljalg, Urmas (2023): UNITE USEARCH/UTAX release for Fungi. Version 18.07.2023. UNITE Community. **https://doi.org/10.15156/BIO/2938083**

unite vs uchime

Reference dataset = celý UNITEto na em generuju chiméry = ást UCHIME datasetu (UCHIM

```
intersection = uchime_ids_set & unite_ids_set
print(len(intersection))
diff = uchime_ids_set - unite_ids_set
print(len(diff))
```

60609

100726

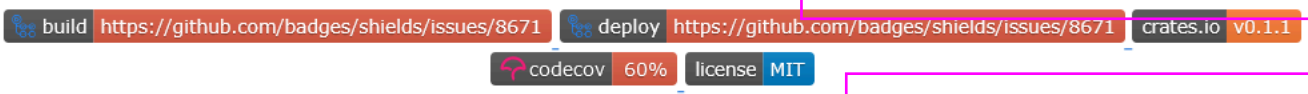
Creation of reads for Simera:

README MIT license

Hyperex

Hyperex je napsanej v Rustu a hází chyby, kdyžnajd

Takže pokud to nkdo chce spustit, tak potebujehy



Možná je lepší použít nco jinýho nebo zto

About

HyperEx (pronounced "Hyper Ex" for Hypervariable region Extractor) is a tool that extracts 16S ribosomal RNA (rRNA) hypervariable region based on a set of primers. By default when no option is specified, hyperex extracts all hypervariable region from the supplied sequences assuming 16S rRNA sequences. To do this it has a set of built-in primer sequences which are universal 16S primers sequences. Nevertheless, the user can choose to specify the wanted region by specifying the `--region` option or by providing the primer sequences using `--forward-primer` and `--reverse-primer`. The `--region` option takes only the region names like "v1v2" or "v4v5" while the `--forward-primer` and `--reverse-primer` takes only the sequences which can contains IUPAC ambiguities.
For more than one needed region, one can use multiple time the `--region`, `--forward-primer`, `reverse-primer` options to specify the wanted region. Theses option takes only one argument, but can be repeat multiple time (see Examples below).

For more praticability, the user can also provide a supplied file containing primer sequences to extract the wanted region using the `--region` option. The primer sequences file should be a no header comma separated value file like:

Primery z GlobalFungi: --- Simera potebuje amplikony ohranienýprimerem, nic jinýho- to

	homogenized primer names	primers sequences
0	18S-F/5.8S-R	GTAAAAGTCGTAACAAGGTTTC/GTTCAAAGAYTCGATGATTC
1	5.8S_Fun/ITS4_Fun	AACTTTYRRCAAYGGATCWCT/AGCCTCCGCTTATTGATATGC
2	58A2F/ITS4	ATCGATGAAGAACGCAG/TCCTCCGCTTATTGATATGC
3	fITS7/ITS4	GTGARTCATCGAATCTTTG/TCCTCCGCTTATTGATATGC
4	fITS9/ITS4	GAACGCAGCRAAIIGYGA/TCCTCCGCTTATTGATATGC
5	FSeq/RSeq	ATGCCTGTTTGAGCGTC/CCTACCTGATTTGAGGTC
6	gITS7/ITS4	GTGARTCATCGARTCTTTG/TCCTCCGCTTATTGATATGC
10	gITS7/ITS4ngs	GTGARTCATCGARTCTTTG/TCCTSCGCTTATTGATATGC
11	gITS7ngs/ITS4ngsUni	GTGARTCATCRARTYTTTG/CCTSCSCTTANTDATATGC
12	ITS1/ITS2	TCCGTAGGTGAACCTGCGG/GCTGCGTTCTTCATCGATGC

	homogenized primer names	primers sequences
13	ITS1/ITS4	TCCGTAGGTGAACCTGCGG/TCCTCCGCTTATTGATATGC
14	ITS1/qITS2*	CTCCGTAGGTGAACCTGCGG/TTYGCTGYGTTCTTCATCG
15	ITS1-30F/ITS1-217R	GTCCCTGCCCTTTGTACACA/TTTCGCTGCGTTCTTCATCG
16	ITS1F/58A2R	CTTGGTCATTTAGAGGAAGTAA/CTGCGTTCTTCATCGAT
17	ITS1F/ITS2	CTTGGTCATTTAGAGGAAGTAA/GCTGCGTTCTTCATCGATG
18	ITS1F/ITS3	CTTGGTCATTTAGAGGAAGTAA/GCATCGATGAAGAACGCAG
19	ITS1F/ITS4	CTTGGTCATTTAGAGGAAGTAA/TCCTCCGCTTATTGATATGC
20	ITS1F/LR6	CTTGGTCATTTAGAGGAAGTAA/CGCCAGTTCTGCTTACC
21	ITS1F_KYO1/ITS2_KYO1	CTHGGTCATTTAGAGGAASTAA/CTRYGTTCTTCATCGDT
22	ITS1F_KYO1/ITS2_KYO2	CTHGGTCATTTAGAGGAASTAA/TTYRCTRCGTTCTTCATC
23	ITS1F_KYO2/ITS2_KYO2	TAGAGGAAGTAAAAGTCGTAA/TTYRCTRCGTTCTTCATC
25	ITS1FI2/5.8S	GAACCWGCGGARGGATCA/CGCTGCGTTCTTCATCG
26	ITS1FI2/ITS2	GAACCWGCGGARGGATCA/GCTGCGTTCTTCATCGATGC
27	ITS1Fngs/ITS2	GGTCATTTAGAGGAAGTAA/GCTGCGTTCTTCATCGATGC
28	ITS1ngs/ITS2	TCCGTAGGTGAACCTGC/GCTGCGTTCTTCATCGATGC
31	ITS2F/ITS2R	GCATCGATGAAGAACGC/CCTCCGCTTATTGATATGC
32	ITS3/ITS4	GCATCGATGAAGAACGCAGC/TCCTCCGCTTATTGATATGC
35	ITS3_KYO2/ITS4	GATGAAGAACGYAGYRAA/TCCTCCGCTTATTGATATGC
36	ITS3_KYO2/ITS4_KYO3	GATGAAGAACGYAGYRAA/CTBTTVCCKCTTCACTCG
47	ITS5/5.8S_fungi	GGAAGTAAAAGTCGTAACAAGG/CAAGAGATCCGTTGTTGAA
48	ITS5/ITS2	GGAAGTAAAAGTCGTAACAAGG/GCTGCGTTCTTCATCGATG
49	ITS5/ITS2_KYO2	GGAAGTAAAAGTCGTAACAAGG/TTYRCTRCGTTCTTCATC
50	ITS5/ITS4	GGAAGTAAAAGTCGTAACAAGG/TCCTCCGCTTATTGATATGC
51	ITS7o/ITS4	GTGAATCATCRAATYTTTG/TCCTCCGCTTATTGATATGC
52	ITS86F/ITS4	GTGAATCATCGAATCTTTGAA/TCCTCCGCTTATTGATATGC
53	ITS86F/ITS4-Tul	GTGAATCATCGAATCTTTGAA/CCGCCAGATTCACACATTGA
54	ITS9/ITS4	GCATTAGAAACTGCTCGTAATG/TCCTCCGCTTATTGATATGC
55	ITS9MUNngs/ITS4ngsUni	TACACACCGCCCGTCG/CCTSCSCTTANTDATATGC

Asi tetinu primer hyperrex v ref. datech z UNITE vbec nenašel - z 1 000 000 sekvencí jsem extrahoval 30 000

Simera

Pro Simeru jsem použil ty amplikony z hypererx s pilepenýma primerama

The Simera software was written as part of my Ph.D. project at the University of Glasgow. This version executes the Simera 2 algorithm which simulates PCR and has been shown to produce more realistic chimeras than other PCR simulation software (http://theses.gla.ac.uk/6801/). The algorithm is described in the accompanying file simera_2_algorithm.pdf.

```
*** OUTPUT FILES ***
```

Simera nešla spustit s moderním gcc - musel jsem to opravit - info v README na githubu

```
- info.txt              Input information.
- all_seqs.fa           All output sequences in FASTA format: full set.
- samp_all_seqs.fa      All output sequences in FASTA format: sampled set.
- good.fa               All good sequences in FASTA format: full set.
- samp_good.fa          All good sequences in FASTA format: sampled set.
- chimeras.fa           All chimeras in FASTA format: full set.
- samp_chimeras.fa      All chimeras in FASTA format: sampled set.
- abund.txt             Output sequence abundances: full set.
- samp_abund.txt        Output sequence abundances: sampled set.
- breaks.txt            Chimera break points: full set.
- samp_breaks.txt       Chimera break points: sampled set.
- parents.fa            Chimera parents in FASTA format: full set.
- samp_parents.txt      Chimera parents in FASTA format: sampled set.
- summary.txt           Summary of output: full set.
- samp_summary.txt      Summary of output: sampled set.
```

info.txt :

Simera akceptuje pouze FASTA co mají MAX 10000 sekvencí-

Version: Simera_v2.1
RNG seed: 1744458363
Input file:
../datasets/uchime_reference_dataset_16_10_2022/2022_10_26_chimera_reference_release/amplicons_from_primers_hyperex/gITS7___ITS4_abund.fasta
Simulated rounds: 25
Lambda: 5.000000e-05
Sampled reads: 10000
Forward primer: GTGARTCATCGARTCTTTG
Reverse primer: TCCTCCGCTTATTGATATGC

summary.txt:

Sequences = 1882
Chimeras = 1655
Max length = 694
Total abundance = 4064940
Chimera abundance = 124954

samp_summary.txt:

Sequences = 391
Chimeras = 169
Max length = 633
Total abundance = 10000
Chimera abundance = 295

Making reference kmers

get_kmers_para_pickles_abund.py

ukládám kmery z reference database na disk v podob: píklad:ge

- groups genus, family ... perm_clust??

ML

Mžu zkusi udlat kmer-clustery z permanent_cluster nebo místo Genus použít Family

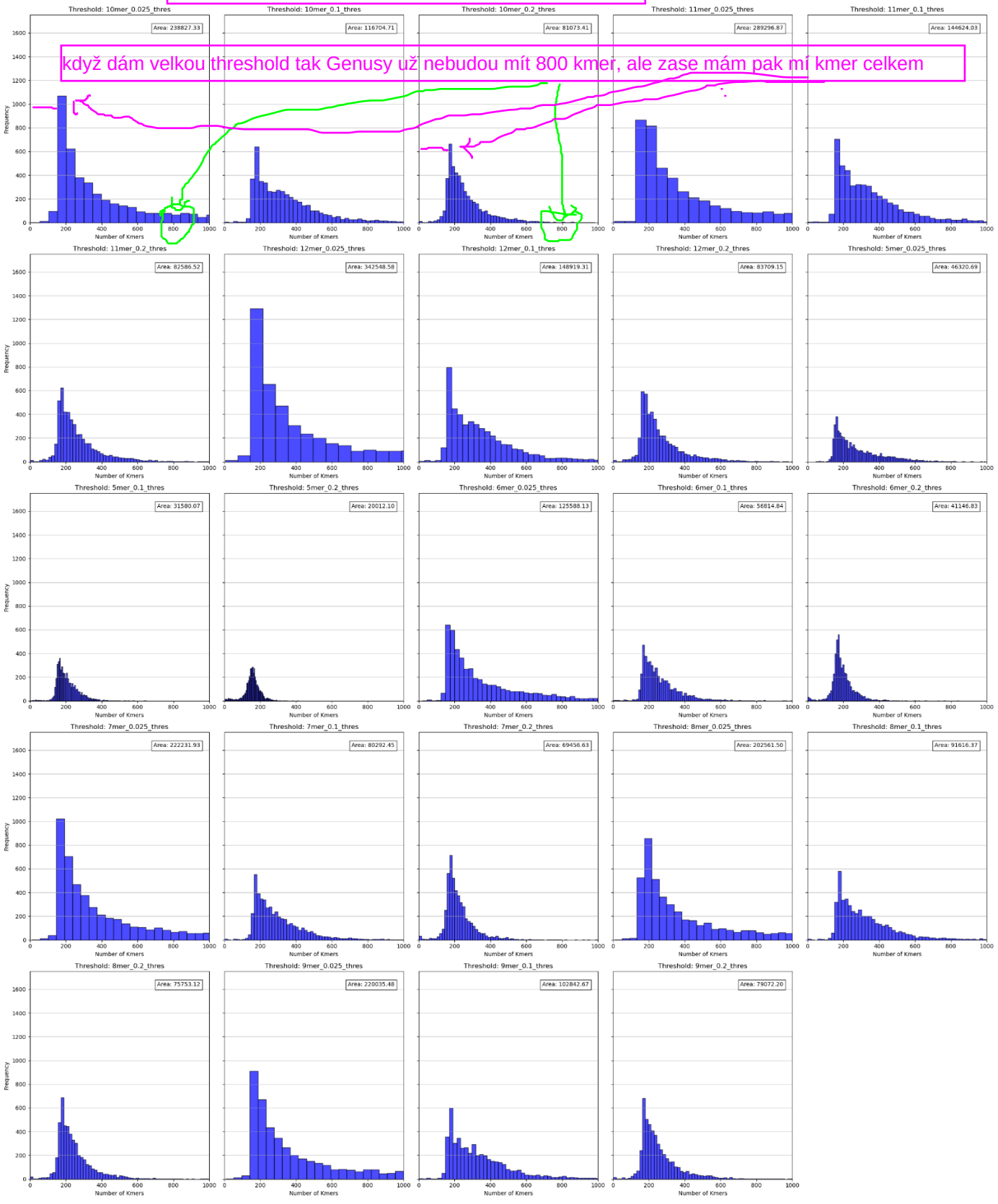
Reference kmers

- discard_kmers_with_low_abund

Number of all kmers in genera:

Vyhazoval jsem kmery který tvoili spodních x (teba 10) % abundance kmer pro genus

To znamená, že:mám g__Russula.npz - to je vytvoený teba z 10 sekvencí s genus == Russula



když dám velkou threshold tak Genusy už nebudou mít 800 kmer, ale zase mám pak mí kmer celkem

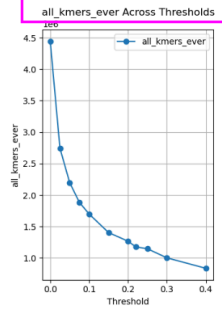
thresholds for discard_kmers_with_low_abund()

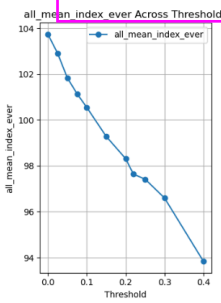
Tady to popíšu co se dje s rostoucí threshold:

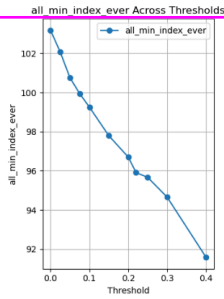
Popisuju na Russula genusu s 10 sekvencema

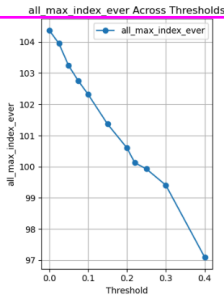
klesá celkový poet kr

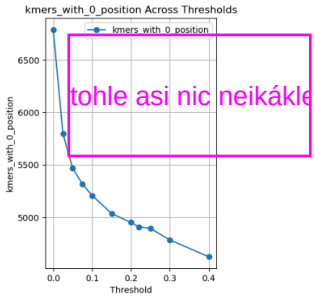
tady v tch tech kledá prmrná pozice všech kmer everto může být tim, že je jede



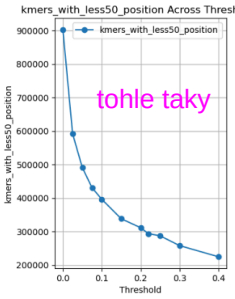




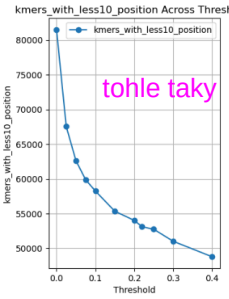




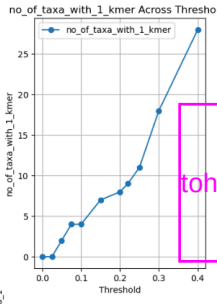
tohle asi nic neikákle



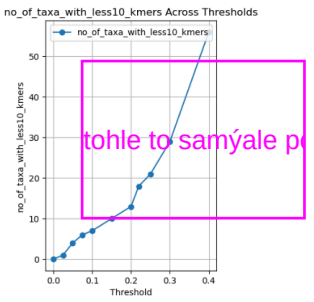
tohle taky



tohle taky

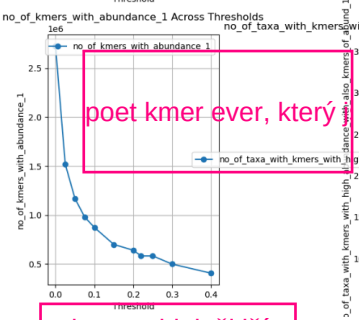


tohle je poet Genuss jen

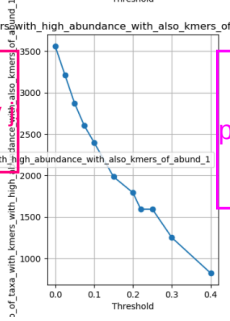


tohle to samýale p

všechny genusy
mají kmery
jinak je vyhodim



poet kmer ever, který

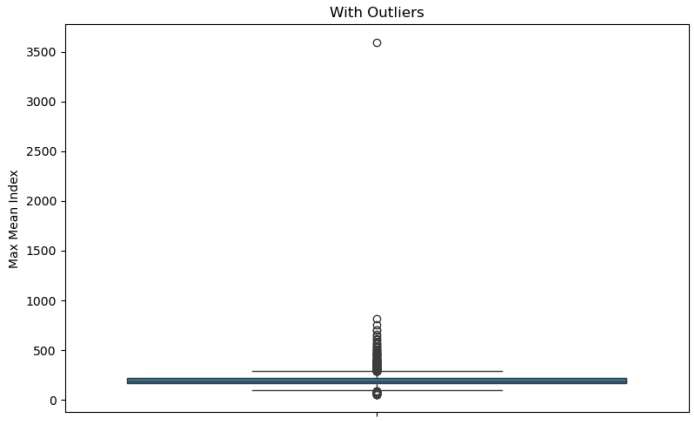


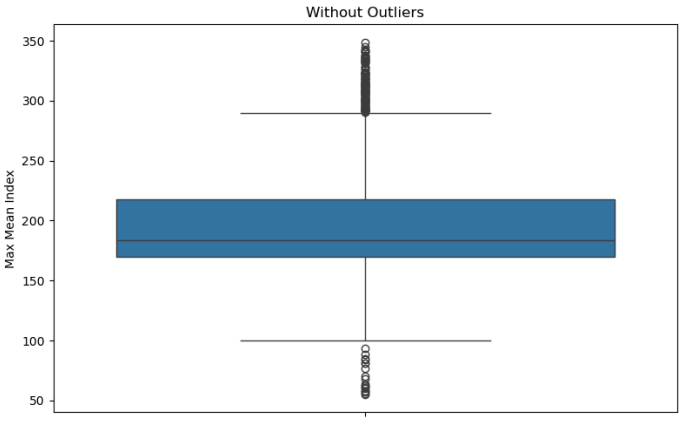
poet Genusco mají ví

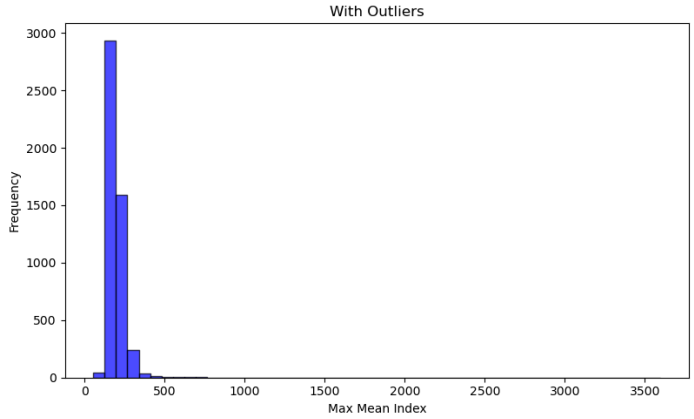
taky dležitý

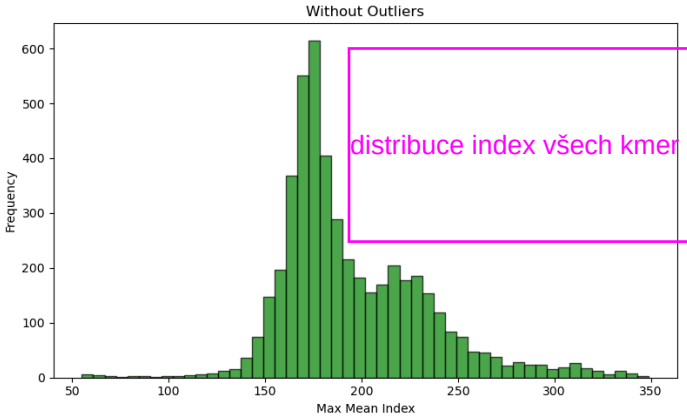
to je to nejduležitjší

maximal index (position) of kmers of taxa:



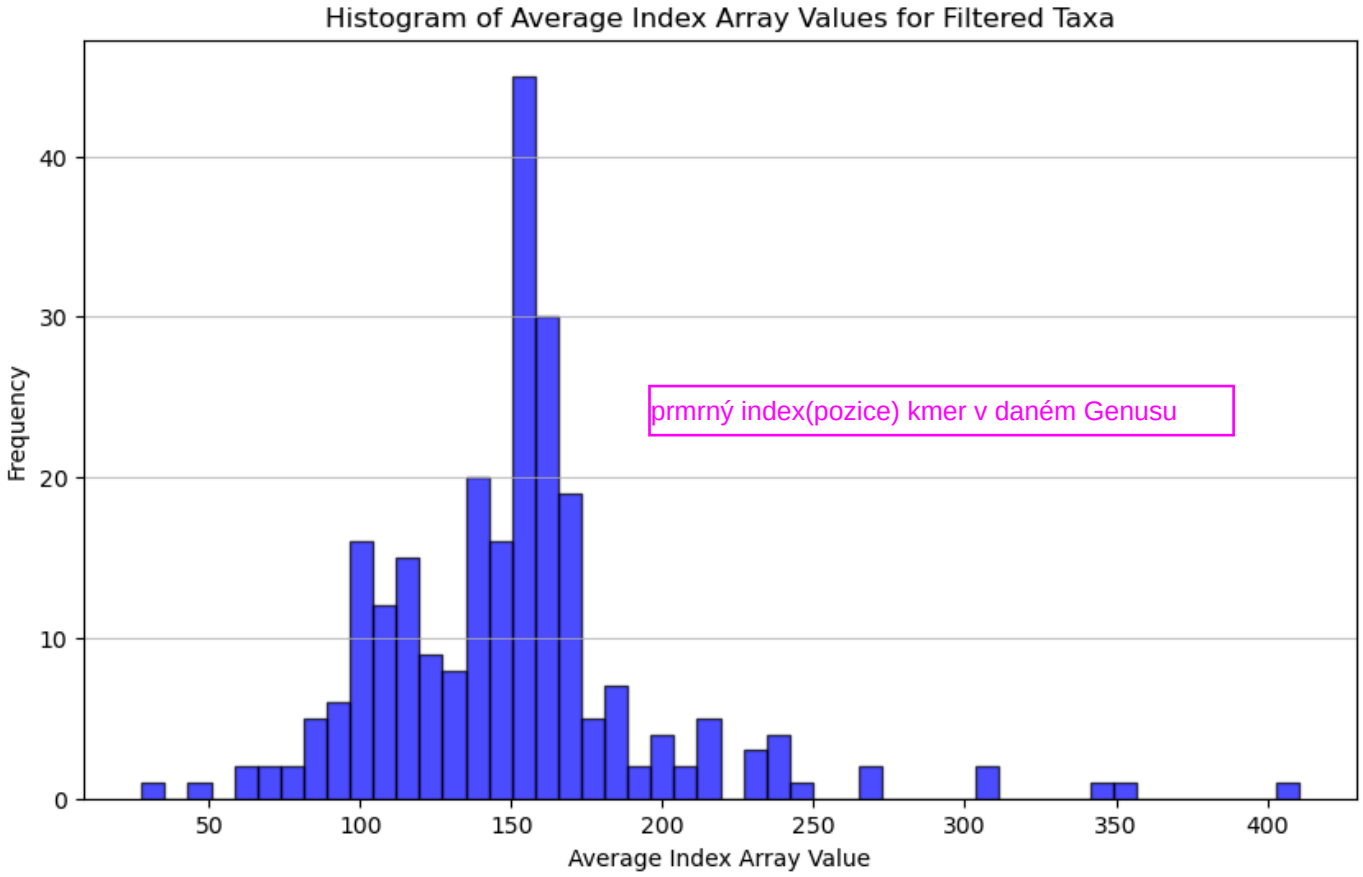






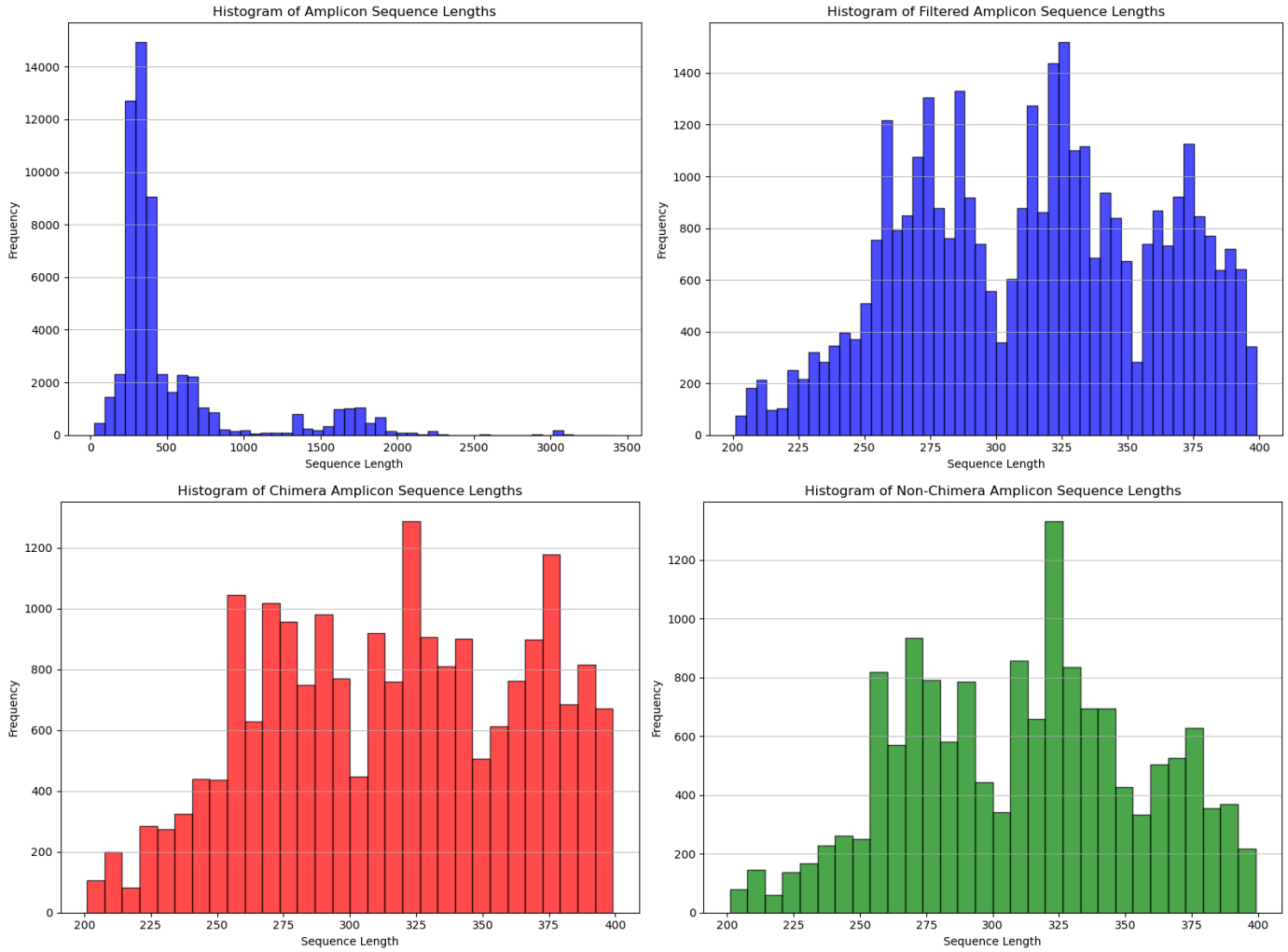
distribuce index všech kmer ever(kdy

averages of of index of kmers of taxa:



Loading Simera output

lengths of all reads from Simera:



- discard all sequences with length < 250 and > 350

Generation of features for ML

Algorithm 1)

tady v tom algoritmu koukám mám sekvenci a tu rozdlim na 5 ástí a každé piadím nejlepší match

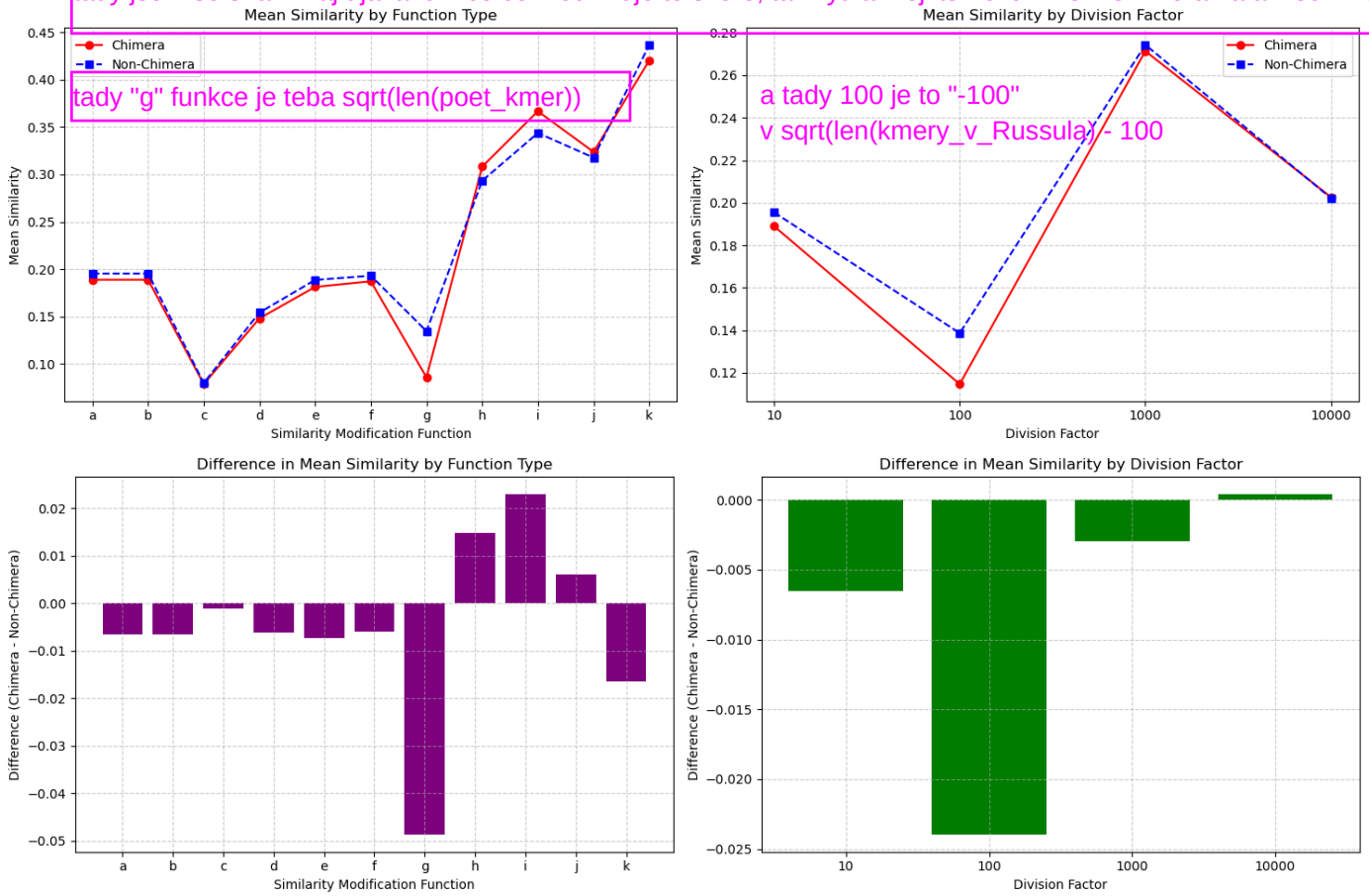
1. split sequence into parts (5,7...)
2. for each part find best match in reference kmers (Jaccard similarity)
3. compute score for the best match group (depends on number of kmers in ref kmer group (Taxa group))
4. compute similarities (Jaccard) between the best-match kmer groups (Taxa group) of all query sequence parts e.g. 1st part best match taxa kmers with 5th part best match taxa kmers

Score modifying functions and division values:

tu podobnost ^^^ zmnim na skóre, kde penalizujuto že teba genus Russula má 10 sekve

a pak srovnám genus match pro 1.ást a genus match 5.ásta s

tady jsem se snažil najít jaká funkce co modifikuje to skóre, tak vytvái nejvtší rozdíl mezi Chimerama a Nechimerama



output snippet:

```
{(7, 'a'): {(1, 2): 0.0,
  (1, 3): 0.0,
  (1, 4): 0.0,
  (1, 5): 0.0,
  (1, 6): 0.0,
  (1, 7): 0.0019267822736030828,
  (2, 3): 0.0345821325648415,
  (2, 4): 0.020338983050847456,
  (2, 5): 0.010282776349614395,
  (2, 6): 0.010282776349614395,
  (2, 7): 0.0,
  (3, 4): 0.024793388429752067,
  (3, 5): 0.024008350730688934,
  (3, 6): 0.024008350730688934,
  (3, 7): 0.003745318352059925,
  (4, 5): 0.019889502762430938,
  (4, 6): 0.019889502762430938,
  (4, 7): 0.0020964360587002098,
  (5, 6): 1.0,
  (5, 7): 0.0010395010395010396,
  (6, 7): 0.0010395010395010396},
 (7, 'b'): {(1, 2): 0.0,
```

```
          (1, 3): 0.0,
          (1, 4): 0.0,
          (1, 5): 0.0,
        ...
```

feature importance:

```
        feature          importance
57      (7, 'c')_(4, 5)  0.028783
40      (7, 'b')_(5, 7)  0.027118
45      (7, 'c')_(1, 5)  0.020988
82      (7, 'd')_(5, 7)  0.019766
124     (7, 'f')_(5, 7)  0.019689
103     (7, 'e')_(5, 7)  0.019568
19      (7, 'a')_(5, 7)  0.018832
42      (7, 'c')_(1, 2)  0.017360
137     (7, 'g')_(3, 4)  0.016106
36      (7, 'b')_(4, 5)  0.015154
133     (7, 'g')_(2, 4)  0.014728
53      (7, 'c')_(3, 4)  0.013619
155     (7, 'h')_(2, 5)  0.012894
15      (7, 'a')_(4, 5)  0.012290
93      (7, 'e')_(2, 6)  0.012179
```

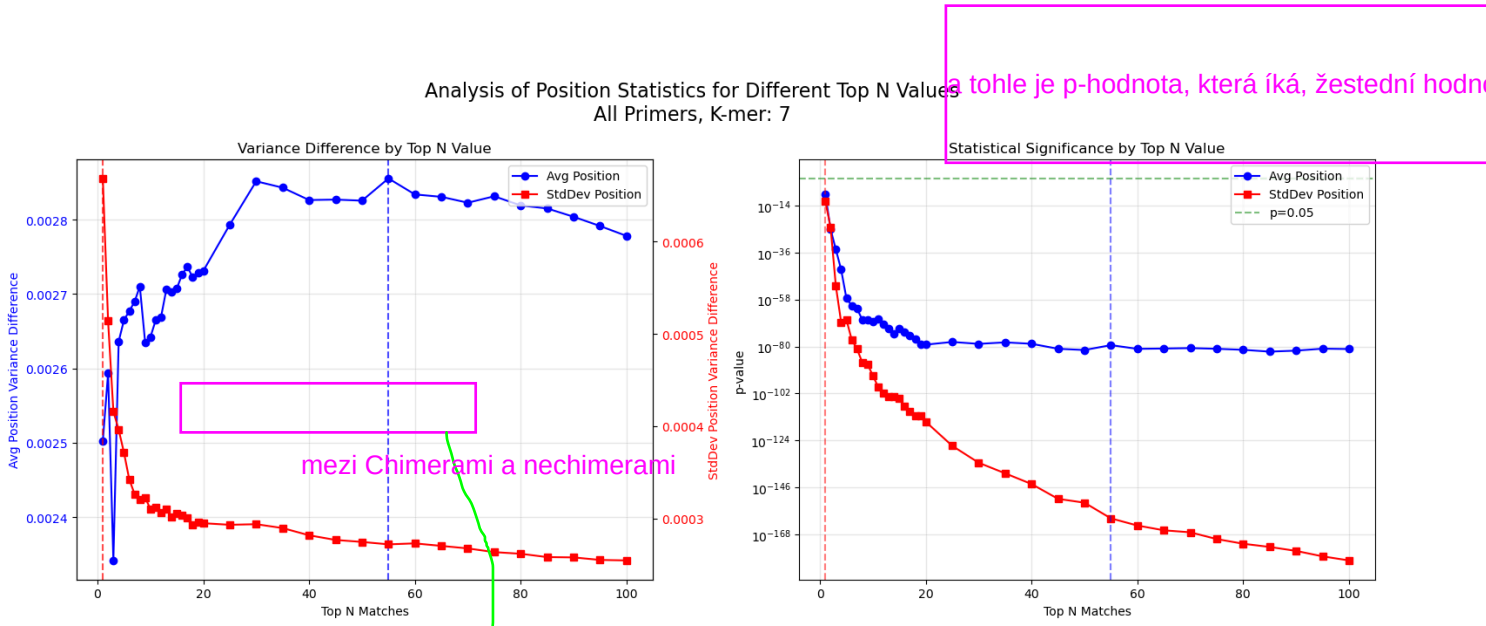
TEST accuracy ~ 0.7 - same as Uchime and Vsearch

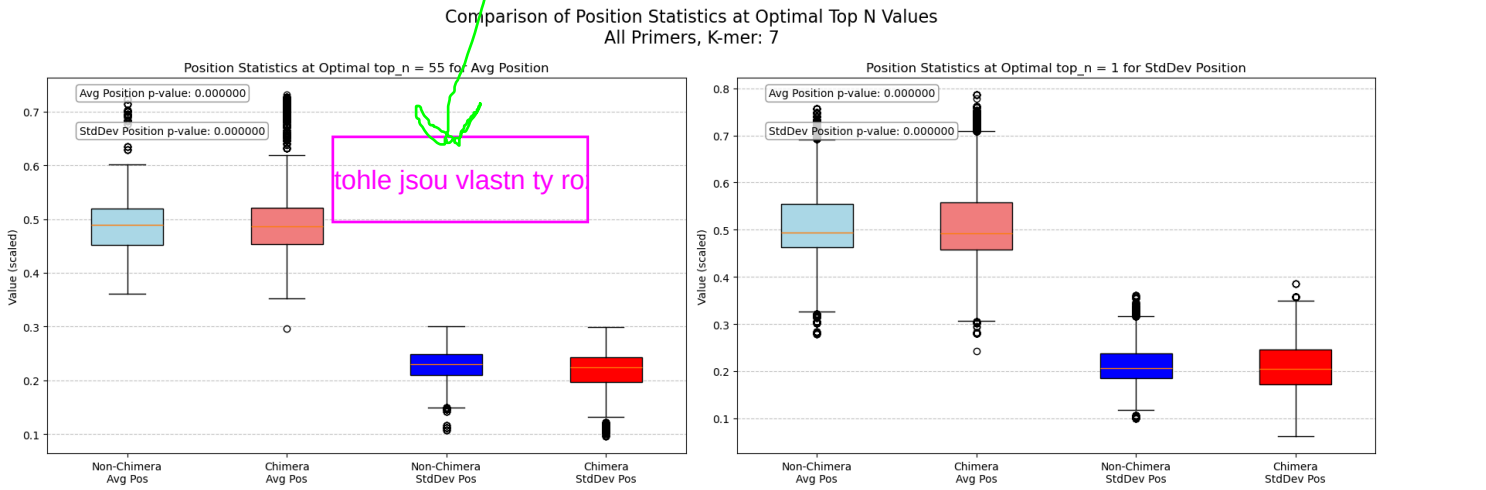
Algorithm 2) Algoritmus 2 :

1. find best match for the query sequence among ref kmer dataset (taxa kmers)
2. get index (positions) of the matching kmers from the query sequence
3. from the matching positions compute average value and standard deviation and scale it by the length of the query sequence
4. find next 99 (or more) best matches and repeat steps 2) and 3) on them

Difference in variance in mean avg_position and stddev_position between Chims and non-chims

Takže ty grafy dole:ty boxploty:První graf s boxploty ukazuje prmrný hodnoty index matching kmer všech NeChime





feature importance

	feature	importance
1	stddev_pos_1	0.020924
71	stddev_pos_100	0.018174
18	avg_pos_10	0.018159
57	stddev_pos_65	0.017941
0	avg_pos_1	0.017924
3	stddev_pos_2	0.017795
59	stddev_pos_70	0.017753
63	stddev_pos_80	0.017688
10	avg_pos_6	0.017394
61	stddev_pos_75	0.017323
7	stddev_pos_4	0.017302
6	avg_pos_4	0.017111
16	avg_pos_9	0.016948
67	stddev_pos_90	0.016561
69	stddev_pos_95	0.016252
47	stddev_pos_40	0.015617
11	stddev_pos_6	0.015482

```
5       stddev_pos_3    0.015318
45      stddev_pos_35   0.015293
8       avg_pos_5       0.015196
```

TEST accuracy on 20-fold CV:

Best parameters: {'max_depth': 25, 'min_samples_split': 2, 'n_estimators': 100}
Best CV accuracy: 0.9557
Best model test accuracy: 0.9587