Assignment 3 - Exercise 1

In [1]:
```python
import subprocess

import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import roc_auc_score, roc_curve
```

Define below two functions for loading the data and scoring the algorithm

In [2]:
```python
def load_data(filename:str):
    """
    Helper function that loads data from storage given
    a filename that contains a full path to the file.
    Assumes that each datapoints is stored as individual line
    in the file.
    """
    data_list = []
    with open(filename, 'r') as f:
        for line in f:
            data_list.append(line[:-1])
    return data_list

# Load the testdata for English and Tagalog
english_test = load_data('english.test')
tagalog_test = load_data('tagalog.test')
```

In [3]:
```python
def get_scores(train_name, test_name, seq_length=10, r=4):
    """
    Run the Negative Selection algorithm implemented in Java.
    This issues a system call to a subprocess with the
    arguments needed to run the Java program.

    PARAMS
    ======
    train_name: The file name (full path) to the training set
    test_name: The file name (full path) to the test set
    seq_length: The length of the sequences in the sets
    r: Parameter r of the Negative Selection algorithm

    RETURNS
    =======
    The score for each of the datapoints in the testset
    """
    # Define the command to run the algorithm with Java
    run_command = \
        f"java -jar negsel2.jar -self {train_name} " \
        f"-n {seq_length} -r {r} -c -l < {test_name}"
    # Issue call to subprocess to run the command
    results = subprocess.getoutput(run_command)
    # Convert the results to numpy array of floats
    return np.array([float(r) for r in results.split('\n')])
```

Exercises 1 & 2

Below is the code to answer exercises 1 and 2.

In [4]:
```
# Collect the score for the best r value
best_score = 0
best_r = None

# Iterate through all r values in range 1 to 10
for r in range(1,10):

    # Train the algorithm on english data, and test on tagalog an
d english
    testresults_tagalog = get_scores(
        'english.train', 'tagalog.test', r=r)
    testresults_english = get_scores(
        'english.train', 'english.test', r=r)

    # Concatenate the results from both languages
    testresults = np.concatenate(
        (testresults_tagalog, testresults_english))

    # Create boolean labels for both datasets
    # (1 for tagalog, 0 for english)
    labels = np.zeros(testresults.shape[0], dtype=bool)
    labels[:testresults_tagalog.shape[0]] = True

    # Compute the ROC AUC score for this setting
    ras = roc_auc_score(labels, testresults)

    if ras > best_score:
        best_r = r
        best_score = ras

    print(f'ROC AUC Score for r={r}: {ras:.4f}')

print('\nThe best score is observed for r={} with {:.4f}'\
      .format(best_r, best_score))
```

```
ROC AUC Score for r=1: 0.5435
ROC AUC Score for r=2: 0.7396
ROC AUC Score for r=3: 0.8311
ROC AUC Score for r=4: 0.7916
ROC AUC Score for r=5: 0.7282
ROC AUC Score for r=6: 0.6681
ROC AUC Score for r=7: 0.5907
ROC AUC Score for r=8: 0.5202
ROC AUC Score for r=9: 0.5121

The best score is observed for r=3 with 0.8311
```

We observe that for r=1 the score is pretty low, which can be explained by the fact that it matches too many strings and is thus underfitting.

For r=9 we observe an equally bad score, which makes sense considering we have 10 letter strings and are thus overfitting on the provided training data.

Exercise 3.

Q: The folder `lang` contains strings from 4 other languages. Determine which of these languages can be best discriminated from English using the negative selection algorithm, and for which of the languages this is most difficult. Can you explain your findings?

In [5]:
```
# Set the directory to the languages
languages_dir = 'lang/'
# The names of the individual languages
languages = [
    'hiligaynon.txt',
    'middle-english.txt',
    'plautdietsch.txt',
    'xhosa.txt'
]
```

In order to answer the question, we select the subrange 2 to 6 of the previous tested r values since values outside of this range did not gather good performance. We loop through all of these values and collect the english test results. Inside of each loop, we loop again through all of the four provided alternative languages. The code inside this second for-loop is the same as above: compute the test score for the alternative language, create labels, and compute the ROC score.

In [6]:
```
for r in [2,3,4,5,6]:
    print(f'Computing for r={r}:')

    testresults_english = get_scores(
        'english.train', 'english.test', r=r)

    for language in languages:
        testresults_lang = get_scores(
            'english.train', languages_dir+language, r=r)

        testresults = np.concatenate(
            (testresults_lang, testresults_english))

        labels = np.zeros(testresults.shape[0], dtype=bool)
        labels[:testresults_lang.shape[0]] = True

        ras = roc_auc_score(labels, testresults)

        print('\tROC AUC Score for language "{}": {:.3f}'\
              .format(language, ras))
```

```
Computing for r=2:
        ROC AUC Score for language "hiligaynon.txt": 0.752
        ROC AUC Score for language "middle-english.txt": 0.514
        ROC AUC Score for language "plautdietsch.txt": 0.707
        ROC AUC Score for language "xhosa.txt": 0.852
Computing for r=3:
        ROC AUC Score for language "hiligaynon.txt": 0.840
        ROC AUC Score for language "middle-english.txt": 0.542
        ROC AUC Score for language "plautdietsch.txt": 0.775
        ROC AUC Score for language "xhosa.txt": 0.889
Computing for r=4:
        ROC AUC Score for language "hiligaynon.txt": 0.797
        ROC AUC Score for language "middle-english.txt": 0.534
        ROC AUC Score for language "plautdietsch.txt": 0.753
        ROC AUC Score for language "xhosa.txt": 0.832
Computing for r=5:
        ROC AUC Score for language "hiligaynon.txt": 0.730
        ROC AUC Score for language "middle-english.txt": 0.522
        ROC AUC Score for language "plautdietsch.txt": 0.701
        ROC AUC Score for language "xhosa.txt": 0.765
Computing for r=6:
        ROC AUC Score for language "hiligaynon.txt": 0.671
        ROC AUC Score for language "middle-english.txt": 0.502
        ROC AUC Score for language "plautdietsch.txt": 0.650
        ROC AUC Score for language "xhosa.txt": 0.692
```

For the full written answer, please consult our PDF hand-in.