1.”Did you mean?” Servisi

Açıkçası bu servisi yazarken baya zorlandım. Araştırmalarım sonucu iki algoritmayı ele aldım: Soundex algoritması ve Levenshtein Distance algoitması. İki algoritmada verilen kelimelerin kendi algoritmalarına göre benzerliklerini bulup(sayısal olarak) ortaya bir ölçüm değeri çıkarmaktadır. Biri bunu matrislerle yaparken diğeri harflere numaralar vererek ölçüm yapmaktadır. Bu servisi C# dilinde yazdım.

1.1.Soundex Algoritması

Öncelikle kullandığım metodun kod parçacığı :

```
public static string Soundex(this string exp)
        {
            string deger = String.Empty;
            if (exp.Length > 1)
            {
                deger = exp.Substring(0, 1);
                exp = exp.Substring(1, exp.Length - 1);

                exp = exp.Replace("Ğ", "G");
                exp = exp.Replace("Ü", "U");
                exp = exp.Replace("Ş", "S");
                exp = exp.Replace("Ö", "O");
                exp = exp.Replace("İ", "I");
                exp = exp.Replace("Ç", "C");


                exp = exp.Replace("A", "");
                exp = exp.Replace("E", "");
                exp = exp.Replace("H", "");
                exp = exp.Replace("I", "");
                exp = exp.Replace("O", "");
                exp = exp.Replace("U", "");
                exp = exp.Replace("W", "");
                exp = exp.Replace("Y", "");

                string karakter = "";
                string o_karakter = "";

                for (int i = 0; i < exp.Length; i++)
                {
                    int code = new int();

                    karakter = exp[i].ToString();
                    if ("BFVP".Contains(karakter))
                    {
                        code = 1;
                    }
                    else if ("CGJKQSXZ".Contains(karakter))
                    {
                        code = 2;
                    }
```

```
                else if ("DT".Contains(karakter))
                {
                    code = 3;
                }
                else if ("L".Contains(karakter))
                {
                    code = 4;
                }
                else if ("MN".Contains(karakter))
                {
                    code = 5;
                }
                else if ("R".Contains(karakter))
                {
                    code = 6;
                }
                if (karakter != o_karakter)
                {
                    if (deger.Substring((deger.Length - 1), 1) != code.ToString())
                    {
                        deger += code.ToString();
                    }
                }
                if (deger.Length == 4)
                {
                    break;
                }
                if (karakter != "")
                {
                    o_karakter = karakter;
                }
            }
            int uzunluk = deger.Length;
            for (int j = 0; j < (4 - uzunluk); j++)
            {
                deger += "0";
            }
        }
        return deger;
    }
```

Bu metodu bir class içinde tanımladım ve formdaki textbox'tan girilen kelimelerin bu metodu kullanmasını sağladım. Kodun unit testini gerçekleştirdim. Performans olarak iyi ama benden ötürü bazı işlemleri tam istediğim gibi gerçekleştirmedi. Bunda verileri text dosyasından almakta etkili oldu. Veritabanından veriler alınsa performansının daha iyi olabileceğini düşünüyorum.

Araştırmalarım sonucu iki algoritma inceledim demiştim. İkisi içinde benim en çok anlayabildiğim Soundex algoritması oldu ve bu yüzden onu seçtim. Kelimeler arasındaki yakınlığı harf numalarına göre hesaplama yöntemi benim için daha mantıklıydı. Levenshtein Distance algoritmasını kullanarakta bu servisi yazdım fakat Soundex'te daha doğru sonuçlar elde ettim.

Performans olarak dediğim gibi veritabanı kullanılarak bu servisin daha kolay yazılabileceğini düşünüyorum. Örnek vermek gerekirse Sql Server'ın içinde kendi select sorgularıyla da bu benzerliğin hesaplandığı uygulamaları gördüm.

Fakat bunları denemedim çünkü siz text dosyasından verileri almamızı istemiştiniz. O yöntemde denenebilir.