BLM2642
Bilgisayar Mühendisleri için Diferansiyel Denklemler Ödevi

Ders Yürütücüsü: Mehmet Fatih Amasyalı

Hazırlayanlar: Bora İLCİ & Yiğit KOTAMAN

Öğrenci Numaraları: 21011035 & 21011089

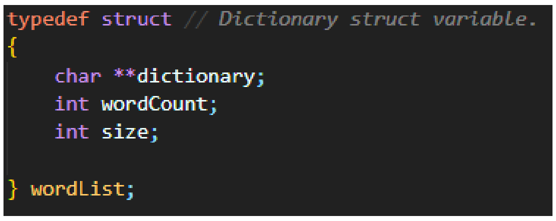
Konu: Optimizasyon algoritmalarını karşılaştırma

Video linki: https://youtu.be/2i0tlejhCxg

A: Metin sınıflandırma

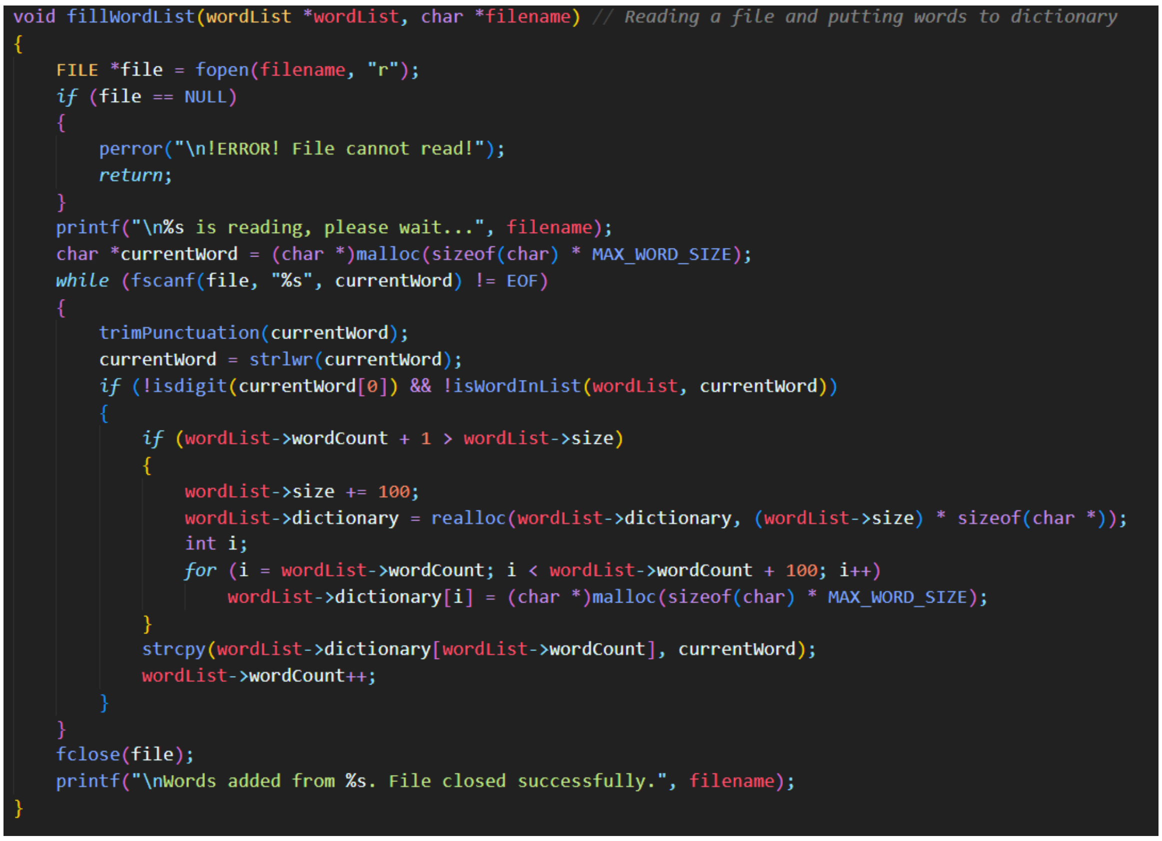
Metin veri setimizde 2 adet metin kümesi bulunuyor.

A sınıfı “Teknoloji”, B sınıfı ise “Film” sınıfıdır. Her iki sınıf için 100’er adet farklı cümle kullanıyoruz ve bu verileri okuyarak 854 adet sözcük içeren sözlük oluşturuyoruz.



```c
typedef struct // Dictionary struct variable.
{
    char **dictionary;
    int wordCount;
    int size;

} wordList;
```



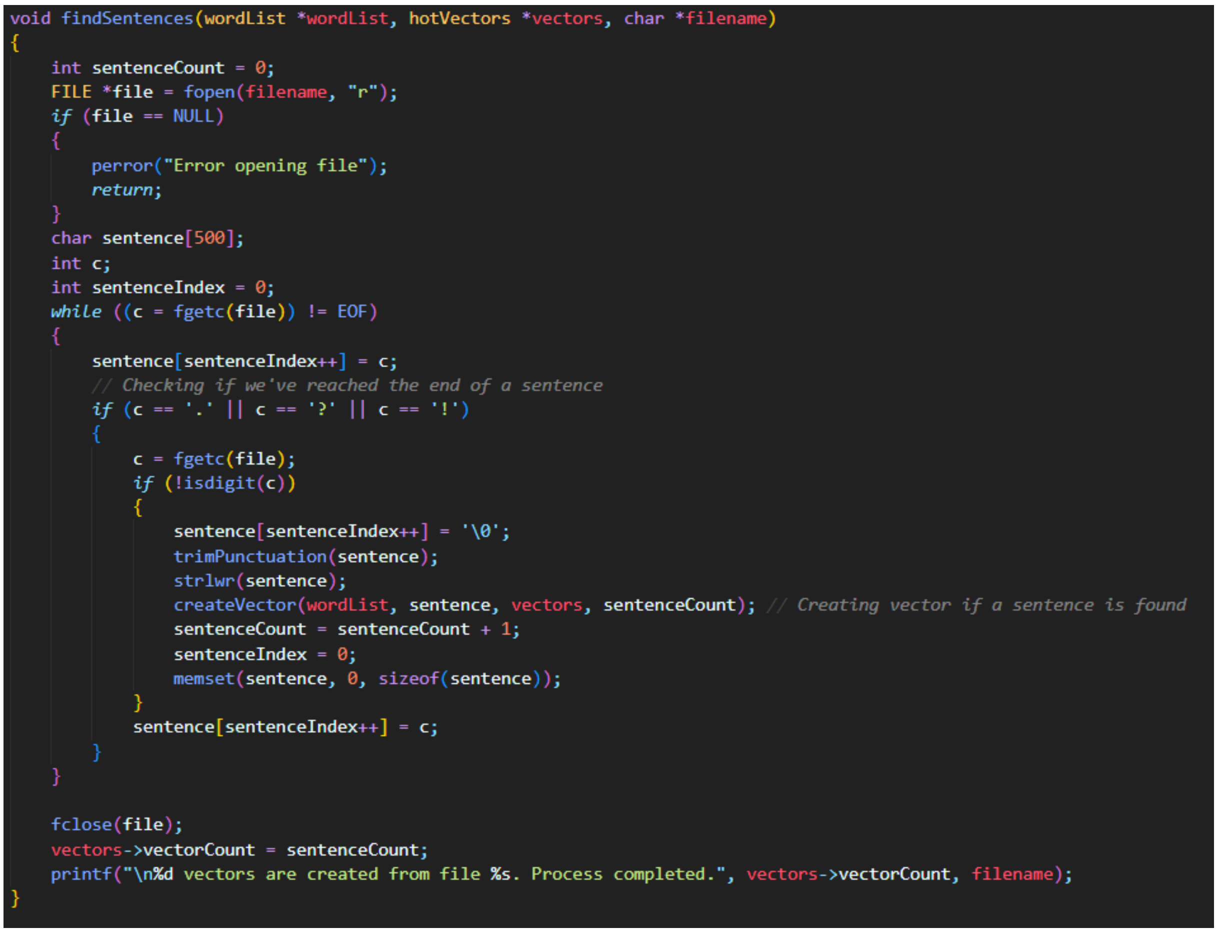
```c
void fillWordList(wordList *wordList, char *filename) // Reading a file and putting words to dictionary
{
    FILE *file = fopen(filename, "r");
    if (file == NULL)
    {
        perror("\n!ERROR! File cannot read!");
        return;
    }
    printf("\n%s is reading, please wait...", filename);
    char *currentWord = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    while (fscanf(file, "%s", currentWord) != EOF)
    {
        trimPunctuation(currentWord);
        currentWord = strlwr(currentWord);
        if (!isdigit(currentWord[0]) && !isWordInList(wordList, currentWord))
        {
            if (wordList->wordCount + 1 > wordList->size)
            {
                wordList->size += 100;
                wordList->dictionary = realloc(wordList->dictionary, (wordList->size) * sizeof(char *));
                int i;
                for (i = wordList->wordCount; i < wordList->wordCount + 100; i++)
                    wordList->dictionary[i] = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
            }
            strcpy(wordList->dictionary[wordList->wordCount], currentWord);
            wordList->wordCount++;
        }
    }
    fclose(file);
    printf("\nWords added from %s. File closed successfully.", filename);
}
```

Kelimeleri **`wordList`** tipinde bir değişken tanımlayarak tutuyoruz.

`fillWordList()` fonksiyonu da verilen .txt dosyasının içerisindeki cümlelerdeki kelimeleri teker teker okuyor ve sözlüğe ekliyor. Eğer sözcük halihazırda sözlükte bulunuyorsa tekrardan eklenmiyor. **`trimPunctuation()`** ve **`strlwr()`** fonksiyonu kelimeleri standardize etmek için kullanılıyor.

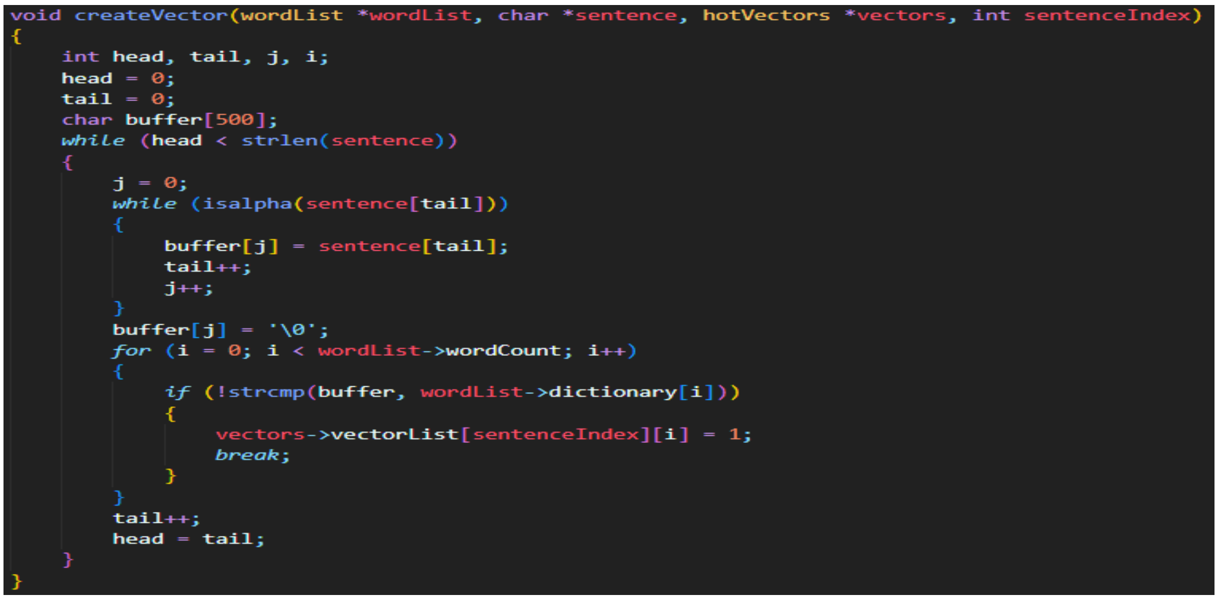
Not: Kaynak kodunun, raporun sonunda paylaşılmıştır. Burada sadece önemli gördüğümüz kod parçaları gösterilmiştir. Ayrıntılı bilgi için kodumuzu inceleyebilirsiniz.

Metinler okunup sözlük oluşturulduktan sonra eğitim için sözlük oluştururken kullandığımız cümlelerin %80'ini seçiyoruz ve o cümlelerden hot vektör oluşturuyoruz.



```c
void findSentences(wordList *wordList, hotVectors *vectors, char *filename)
{
    int sentenceCount = 0;
    FILE *file = fopen(filename, "r");
    if (file == NULL)
    {
        perror("Error opening file");
        return;
    }
    char sentence[500];
    int c;
    int sentenceIndex = 0;
    while ((c = fgetc(file)) != EOF)
    {
        sentence[sentenceIndex++] = c;
        // Checking if we've reached the end of a sentence
        if (c == '.' || c == '?' || c == '!')
        {
            c = fgetc(file);
            if (!isdigit(c))
            {
                sentence[sentenceIndex++] = '\0';
                trimPunctuation(sentence);
                strlwr(sentence);
                createVector(wordList, sentence, vectors, sentenceCount); // Creating vector if a sentence is found
                sentenceCount = sentenceCount + 1;
                sentenceIndex = 0;
                memset(sentence, 0, sizeof(sentence));
            }
            sentence[sentenceIndex++] = c;
        }
    }

    fclose(file);
    vectors->vectorCount = sentenceCount;
    printf("\n%d vectors are created from file %s. Process completed.", vectors->vectorCount, filename);
}
```



```c
void createVector(wordList *wordList, char *sentence, hotVectors *vectors, int sentenceIndex)
{
    int head, tail, j, i;
    head = 0;
    tail = 0;
    char buffer[500];
    while (head < strlen(sentence))
    {
        j = 0;
        while (isalpha(sentence[tail]))
        {
            buffer[j] = sentence[tail];
            tail++;
            j++;
        }
        buffer[j] = '\0';
        for (i = 0; i < wordList->wordCount; i++)
        {
            if (!strcmp(buffer, wordList->dictionary[i]))
            {
                vectors->vectorList[sentenceIndex][i] = 1;
                break;
            }
        }
        tail++;
        head = tail;
    }
}
```

Eğitim için kullanacağımız cümlelerin içerisindeki kelimeleri sözlükte var olup olmadığına bakarak kelime sayısı * 1 boyutlu hot vektör oluşturuyoruz.

Modelimizin amacı: A sınıfı ile alakalı bir metin verildiğinde 1, B sınıfı ile alakalı bir metin verildiğinde -1 çıkışını vermesi

Çıkış fonksiyonumuz: $tanh(w * x)$

w : $\text{öğrenilecek parametreler}$

x : hot vektör

Modelimiz için 5 farklı w başlangıç değeri belirledik:

$w_1 : 0$

$w_2 : 0.02$

$w_3 : -0.02$

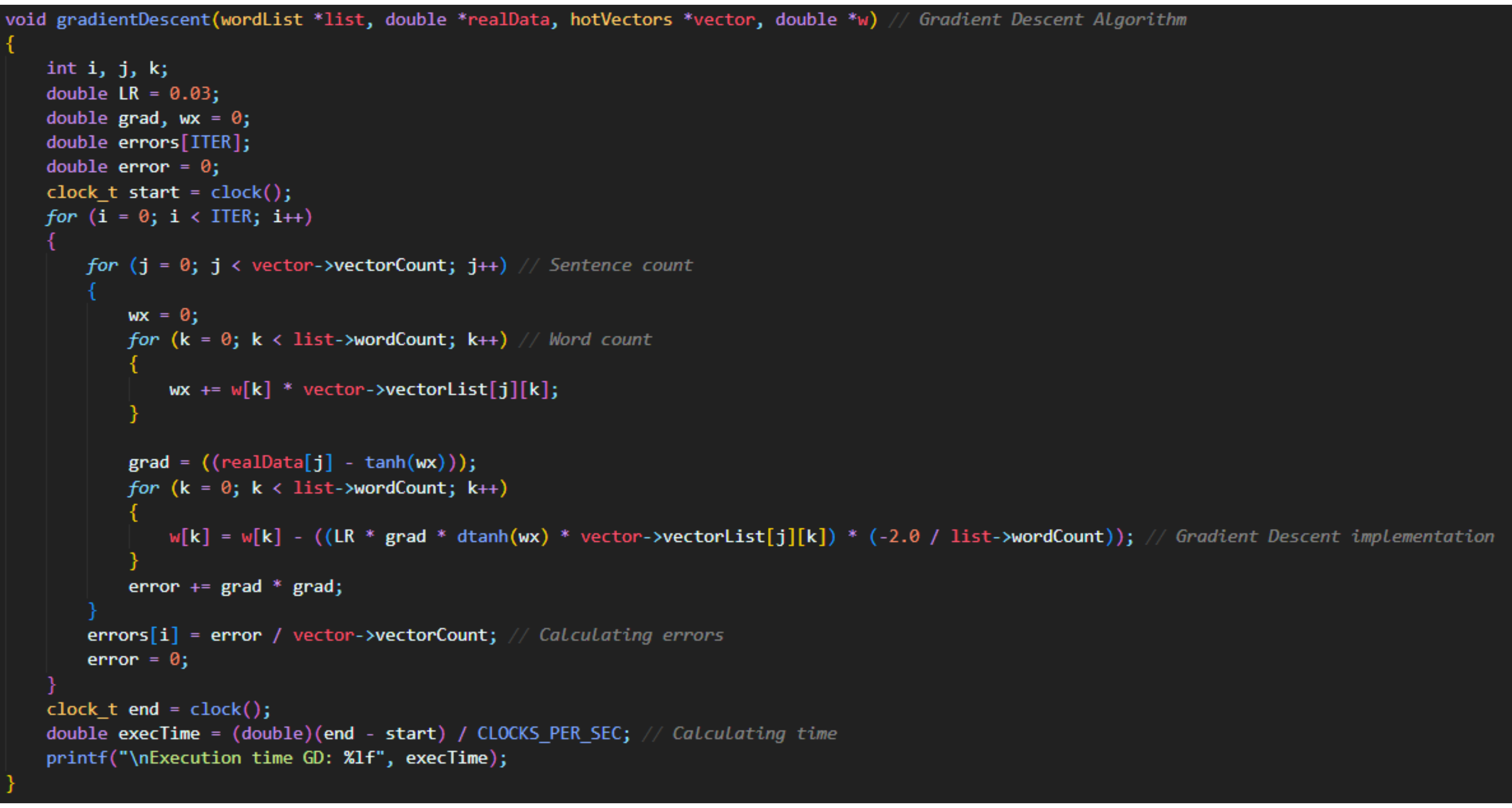
$w_4 : 0.1$

$w_5 : -0.3$

Seçtiğimiz w başlangıç değerlerine göre modelimizi eğittik. Bu işlemi gerçekleştirirken 3 farklı optimizasyon algoritması kullandık:

1) Gradient Descent
2) Stochastic Gradient Descent
3) ADAM

Gradient Descent



```c
void gradientDescent(wordList *list, double *realData, hotVectors *vector, double *w) // Gradient Descent Algorithm
{
    int i, j, k;
    double LR = 0.03;
    double grad, wx = 0;
    double errors[ITER];
    double error = 0;
    clock_t start = clock();
    for (i = 0; i < ITER; i++)
    {
        for (j = 0; j < vector->vectorCount; j++) // Sentence count
        {
            wx = 0;
            for (k = 0; k < list->wordCount; k++) // Word count
            {
                wx += w[k] * vector->vectorList[j][k];
            }

            grad = ((realData[j] - tanh(wx)));
            for (k = 0; k < list->wordCount; k++)
            {
                w[k] = w[k] - ((LR * grad * dtanh(wx) * vector->vectorList[j][k]) * (-2.0 / list->wordCount)); // Gradient Descent implementation
            }
            error += grad * grad;
        }
        errors[i] = error / vector->vectorCount; // Calculating errors
        error = 0;
    }
    clock_t end = clock();
    double execTime = (double)(end - start) / CLOCKS_PER_SEC; // Calculating time
    printf("\nExecution time GD: %lf", execTime);
}
```

Learning rate: 0.03

Iteration count: 1000

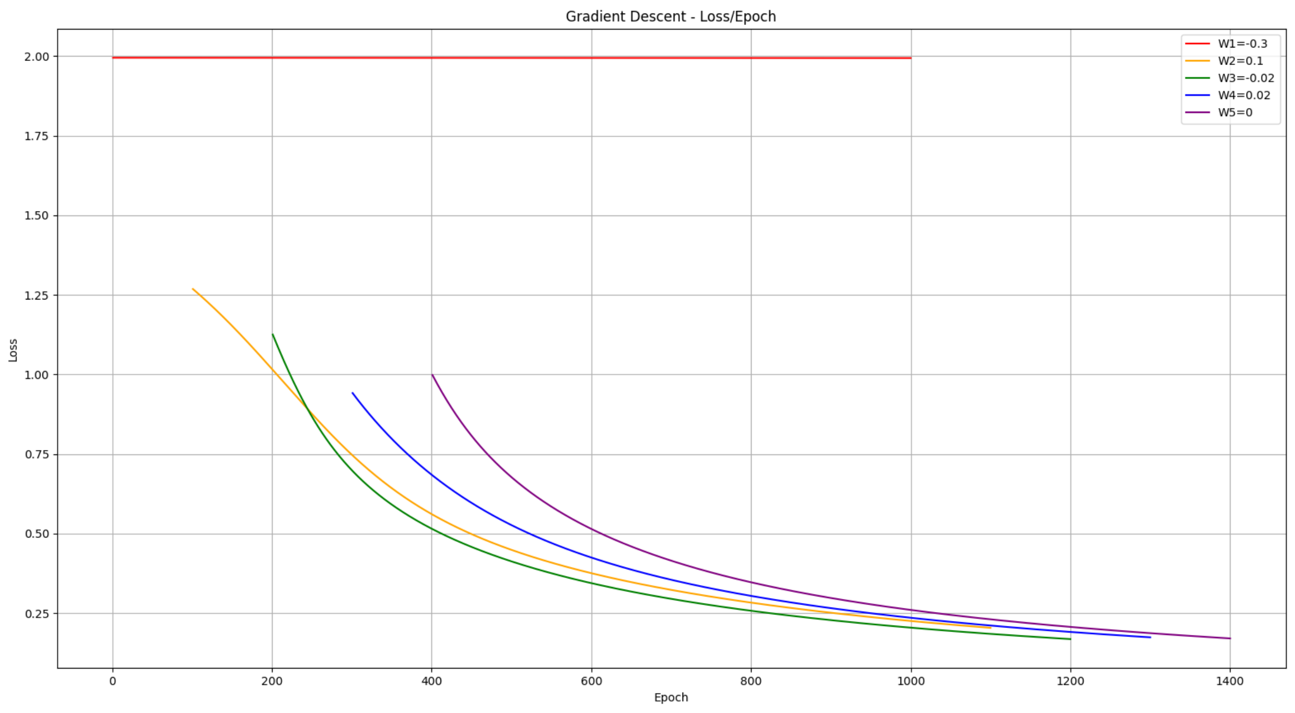
GD algoritmasında parametrelerimizi hataları minimize ederek eğitiyoruz.

MSE: $\frac{1}{N} * \sum(y - \tanh(w * x))^2$

Gradyan: $2(y - \tanh(w * x)) * (-(tanh)'(w * x) * x)$

$$w_{i+1} = w_i - learning_{rate} * \nabla \mathrm{f}$$

Gradient Descent Grafiği (Epoch/Loss)

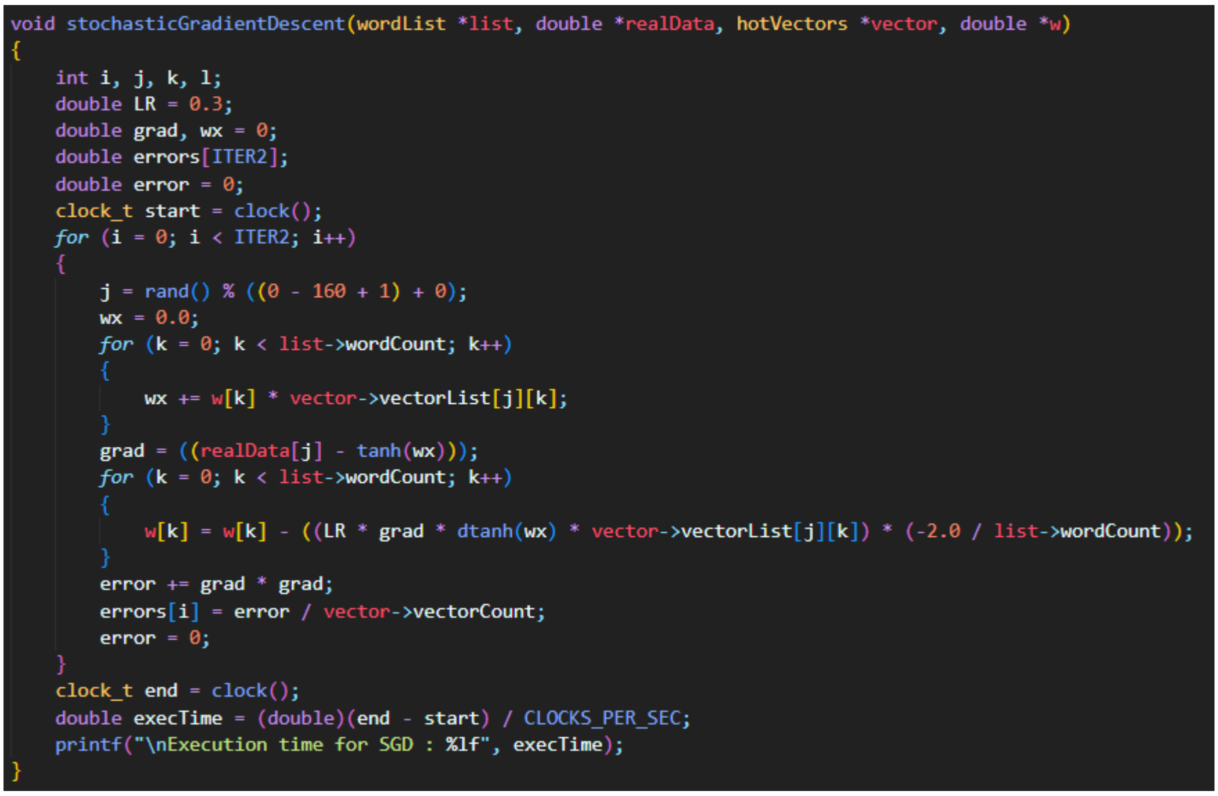


Her parametre için yaklaşık çalışma süresi $\approx 2.3\ sn$

Grafikten aşağıdaki çıkarımlar yapılabilir:

1) Seçilen 4 başlangıç parametresi için algoritma gayet tutarlı bir şekilde hata oranını minimize ediyor.
2) W = -0.3 alındığında model, yaklaşması gereken katsayıya ulaşmıyor. Buradan başlangıç katsayısının etkisini gözlemleyebiliyoruz.

Stochastic Gradient Descent



```c
void stochasticGradientDescent(wordList *list, double *realData, hotVectors *vector, double *w)
{
    int i, j, k, l;
    double LR = 0.3;
    double grad, wx = 0;
    double errors[ITER2];
    double error = 0;
    clock_t start = clock();
    for (i = 0; i < ITER2; i++)
    {
        j = rand() % ((0 - 160 + 1) + 0);
        wx = 0.0;
        for (k = 0; k < list->wordCount; k++)
        {
            wx += w[k] * vector->vectorList[j][k];
        }
        grad = ((realData[j] - tanh(wx)));
        for (k = 0; k < list->wordCount; k++)
        {
            w[k] = w[k] - ((LR * grad * dtanh(wx) * vector->vectorList[j][k]) * (-2.0 / list->wordCount));
        }
        error += grad * grad;
        errors[i] = error / vector->vectorCount;
        error = 0;
    }
    clock_t end = clock();
    double execTime = (double)(end - start) / CLOCKS_PER_SEC;
    printf("\nExecution time for SGD : %lf", execTime);
}
```

Learning rate: 0.3

Iteration count: 10000

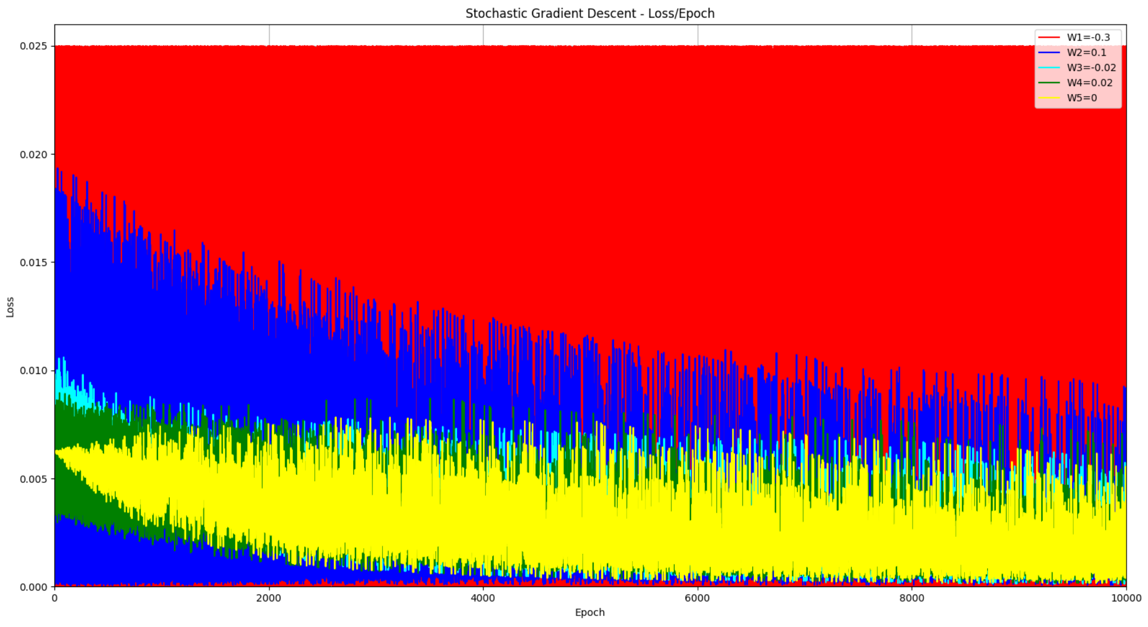
SGD algoritmasında GD’den farklı olarak tüm cümleleri aynı anda değil rastgele bir cümleyi baz alarak eğitimi gerçekleştiriyoruz.

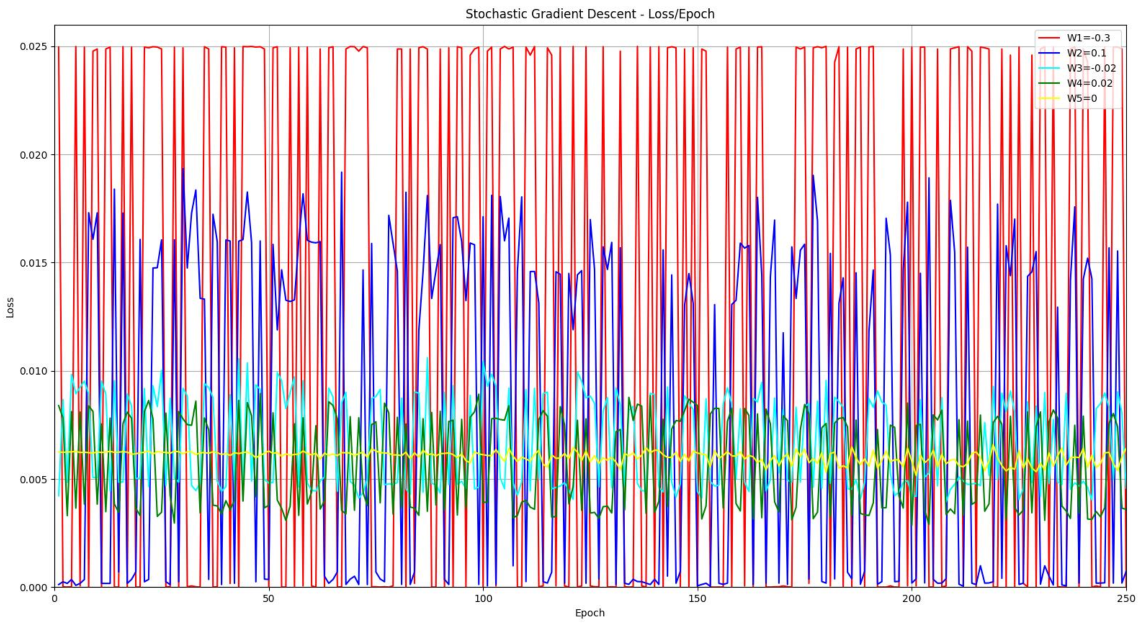
MSE: $\frac{1}{N} * \sum (y - \tanh(w * x))^2$

Gradyan: $2(y - \tanh(w * x)) * (-(tanh)'(w * x) * x)$

$$w_{i+1} = w_i - learning_{rate} * \nabla \mathrm{f}$$

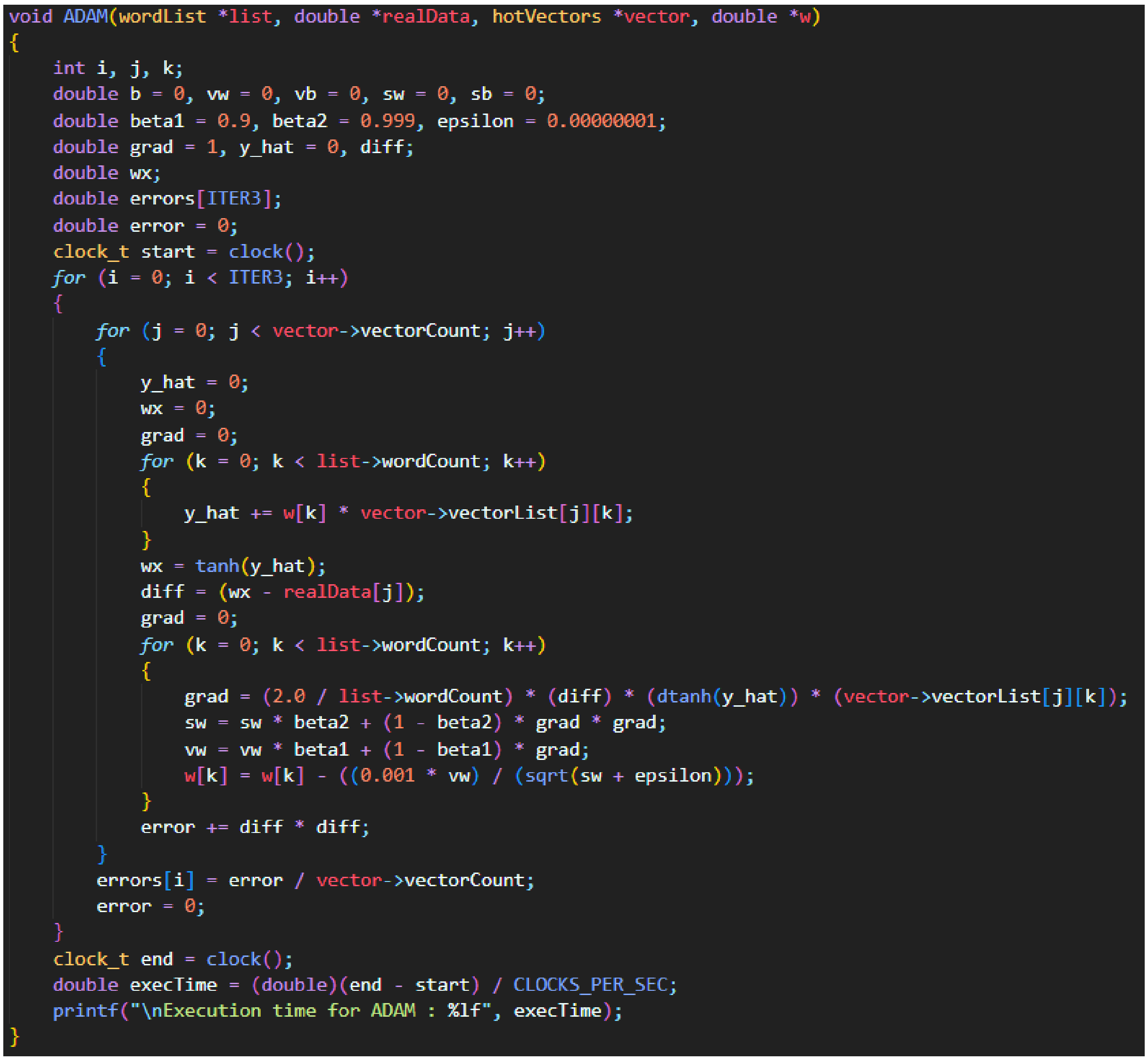
Stochastic Gradient Descent Grafiği (Epoch/Loss)





Not: 2. grafik, algoritmanın daha iyi yorumlanabilmesi için 1. grafiğin yakınlaştırılmış halidir.

ADAM



```
void ADAM(wordList *list, double *realData, hotVectors *vector, double *w)
{
    int i, j, k;
    double b = 0, vw = 0, vb = 0, sw = 0, sb = 0;
    double beta1 = 0.9, beta2 = 0.999, epsilon = 0.00000001;
    double grad = 1, y_hat = 0, diff;
    double wx;
    double errors[ITER3];
    double error = 0;
    clock_t start = clock();
    for (i = 0; i < ITER3; i++)
    {
        for (j = 0; j < vector->vectorCount; j++)
        {
            y_hat = 0;
            wx = 0;
            grad = 0;
            for (k = 0; k < list->wordCount; k++)
            {
                y_hat += w[k] * vector->vectorList[j][k];
            }
            wx = tanh(y_hat);
            diff = (wx - realData[j]);
            grad = 0;
            for (k = 0; k < list->wordCount; k++)
            {
                grad = (2.0 / list->wordCount) * (diff) * (dtanh(y_hat)) * (vector->vectorList[j][k]);
                sw = sw * beta2 + (1 - beta2) * grad * grad;
                vw = vw * beta1 + (1 - beta1) * grad;
                w[k] = w[k] - ((0.001 * vw) / (sqrt(sw + epsilon)));
            }
            error += diff * diff;
        }
        errors[i] = error / vector->vectorCount;
        error = 0;
    }
    clock_t end = clock();
    double execTime = (double)(end - start) / CLOCKS_PER_SEC;
    printf("\nExecution time for ADAM : %lf", execTime);
}
```

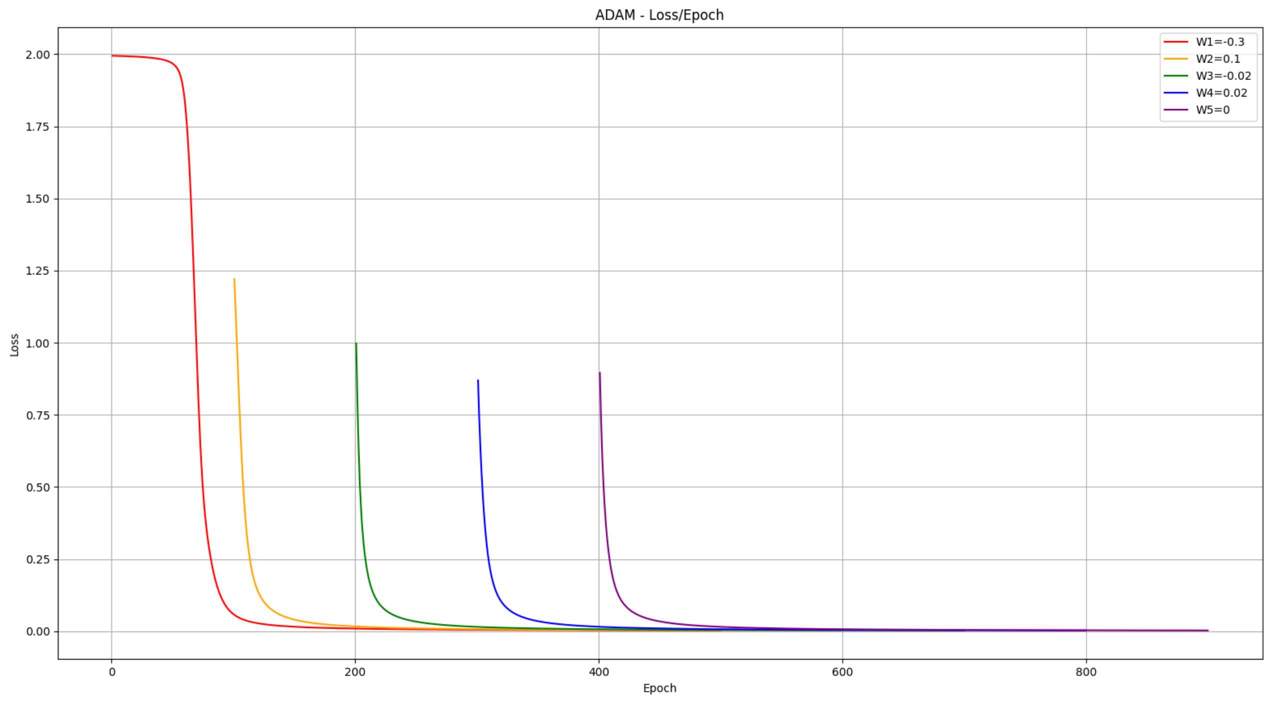
Learning rate (α): 0.001(default)

Iteration count: 500

$$\text{MSE: } \frac{1}{N} * \sum (y - \tanh(w * x))^2$$

$$w_{i+1} = w_i - \alpha * \frac{V_{dw}}{\sqrt{S_{dw} + \varepsilon}}$$

ADAM Grafiği (Epoch/Loss)



Grafiğe bakıldığında açıkça görülüyor ki ADAM algoritması için başlangıç değerinin etkisi diğer algoritmalara kıyasla çok daha az. Özellikle GD ve SGD algoritmalarında w = -0.3 noktasından başlatıldığında istenen değere yakınsamıyorken ADAM istenen sonuca ulaşıyor.

Her parametre için yaklaşık çalışma süresi $\approx 1.7\ sn$

GD – SGD – ADAM Algoritma Karşılaştırması

Aşağıdaki tabloda birbirinden farklı 5 w başlangıç katsayısının her algoritma için başarı oranları gösterilmiştir.

$w_1: 0$

$w_2: 0.02$

$w_3: -0.02$

$w_4: 0.1$

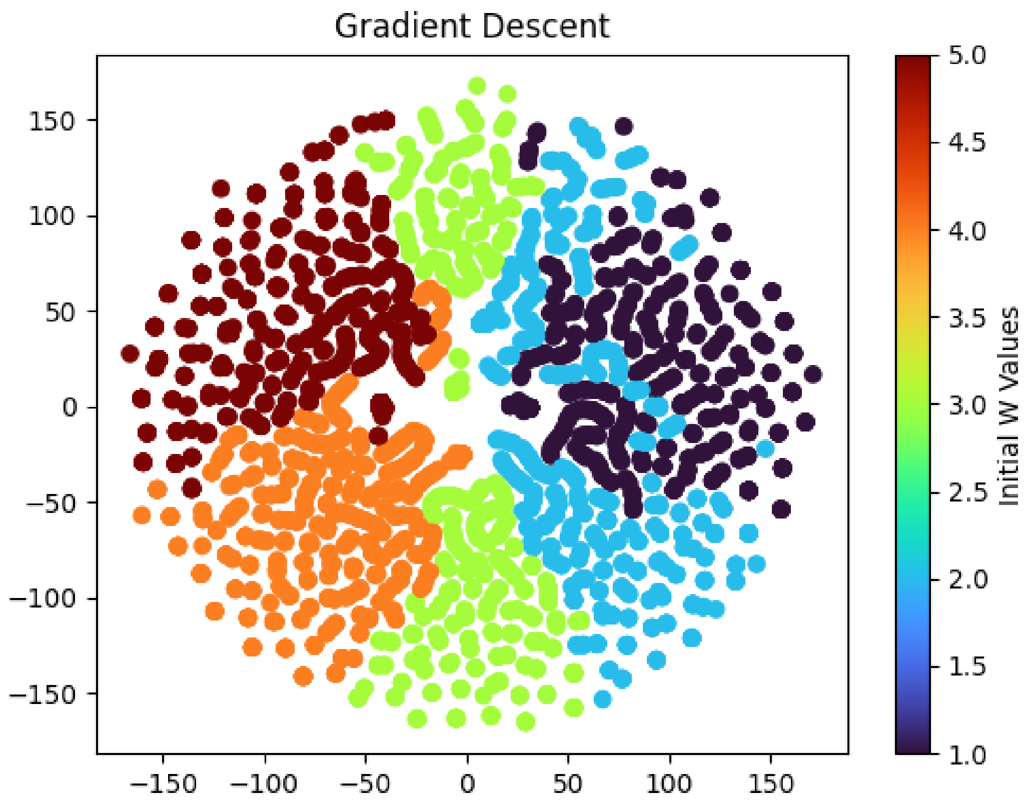
$w_5: -0.3$

Algoritma/Başarı Oranı	w_1	w_2	w_3	w_4	w_5	Süre (s)
GD	%82.5	%82.5	%77.5	%77.5	%50	2.34
SGD	%77.5	%77.5	%72.5	%70	%50	0.14
ADAM	%97.5	%97.5	%100	%97.5	%97.5	1.7

Not: Bu değerler kaynak kodu çalıştırıldığında elde edilebilir.

B: Optimizasyon sürecinin 2 boyutta gösterimi

Gradient Descent in T-SNE



Eğitim sonrası katsayıların dağılımı

$w_1 : 0\ (koyu\ mavi)$

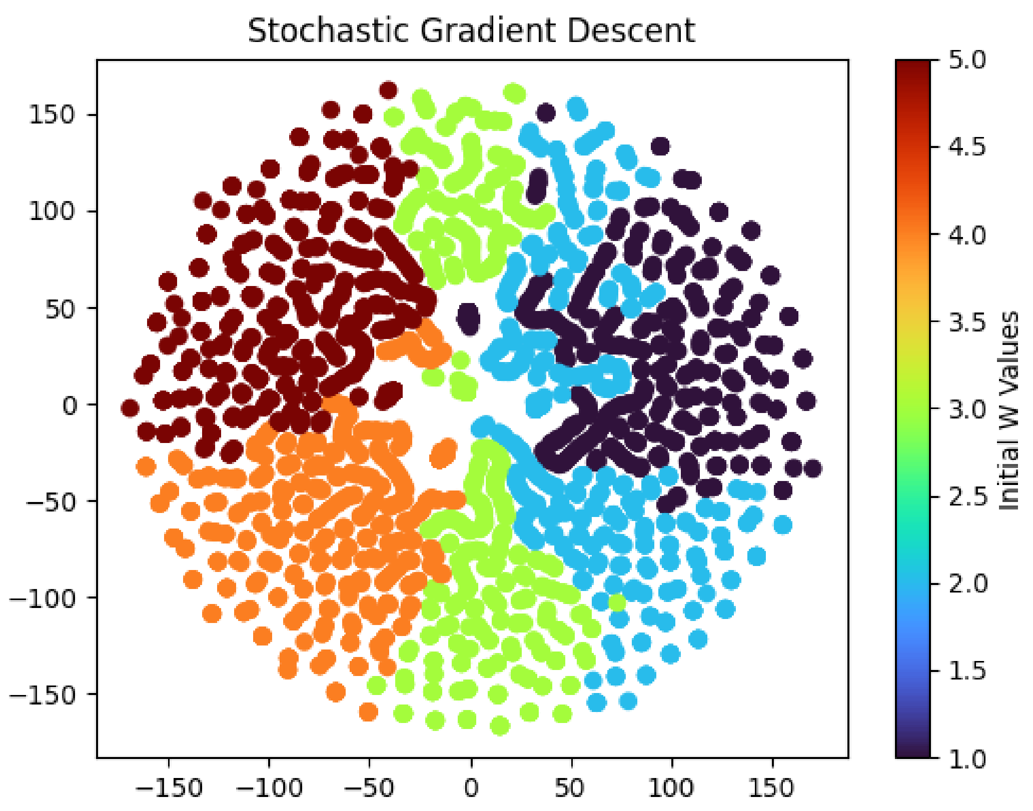
$w_2 : 0.02\ (açık\ mavi)$

$w_3 : -0.02\ (yeşil)$

$w_4 : 0.1\ (turuncu)$

$w_5 : -0.3\ (koyu\ kırmızı)$

Stochastic Gradient Descent in T-SNE



Eğitim sonrası katsayıların dağılımı

$w_1: 0\ (koyu\ mavi)$

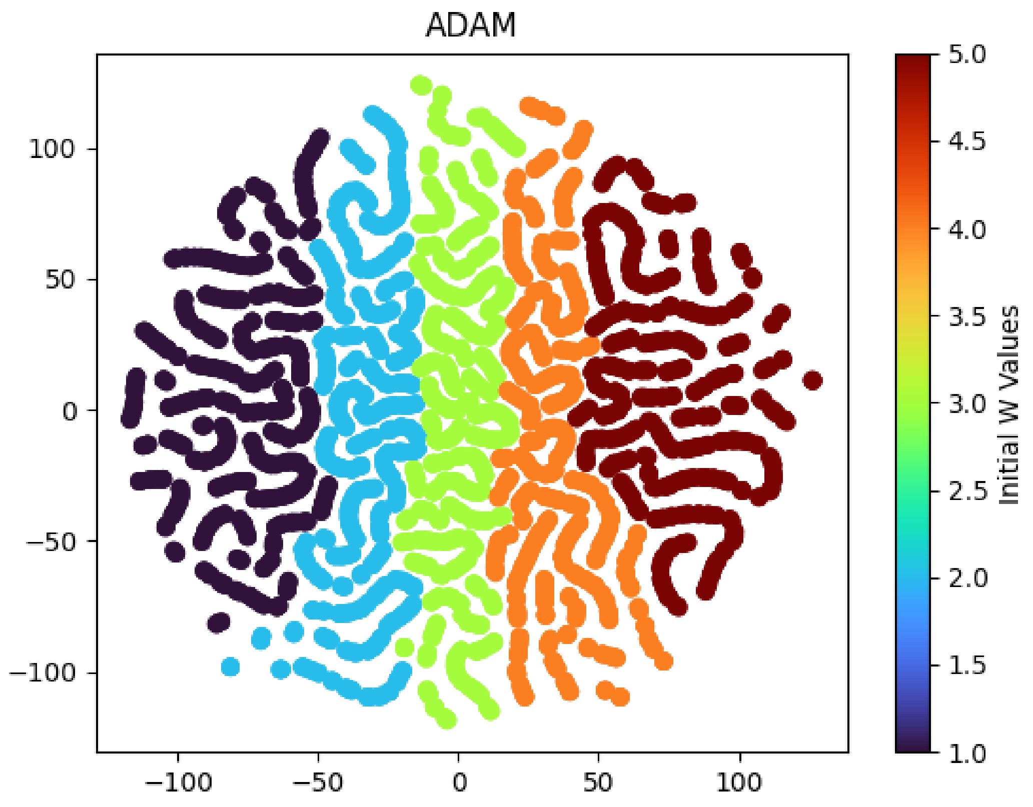
$w_2: 0.02\ (açık\ mavi)$

$w_3: -0.02\ (yeşil)$

$w_4: 0.1\ (turuncu)$

$w_5: -0.3\ (koyu\ kırmızı)$

ADAM in T-SNE



Eğitim sonrası katsayıların dağılımı

$w_1: 0\ (koyu\ mavi)$

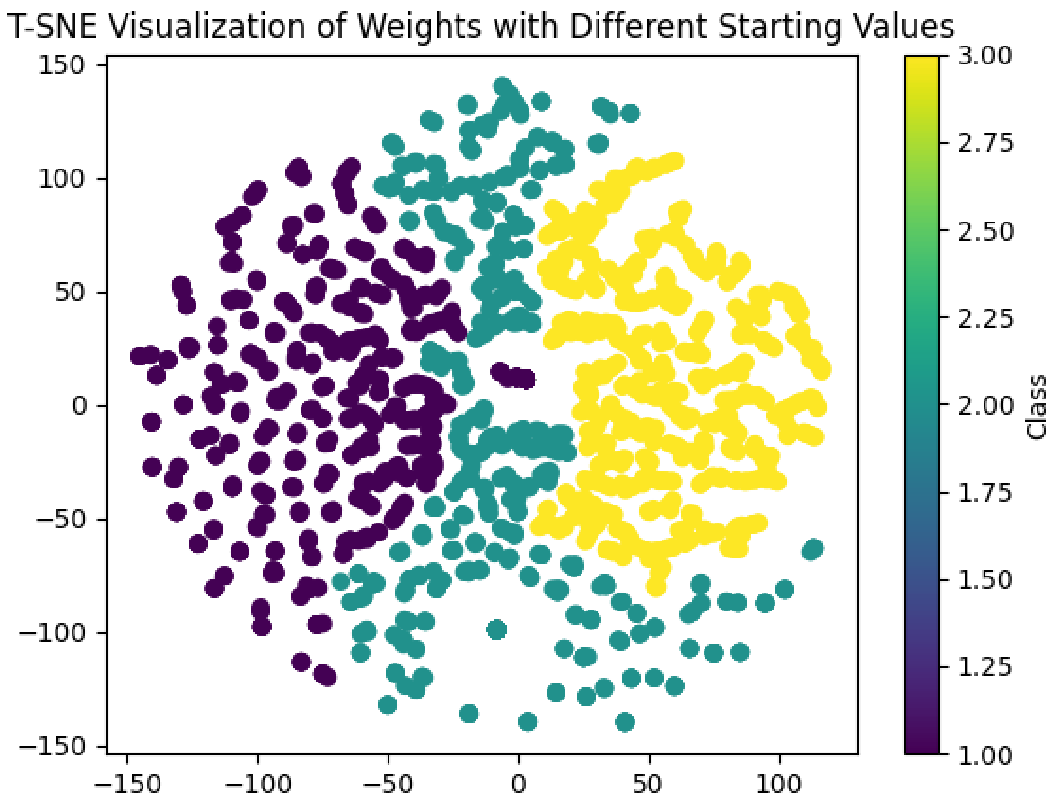
$w_2: 0.02\ (açık\ mavi)$

$w_3: -0.02\ (yeşil)$

$w_4: 0.1\ (turuncu)$

$w_5: -0.3\ (koyu\ kırmızı)$

GD – SGD – ADAM in same T-SNE



1) *Gradient Descent (koyu mavi)*
2) *Stochastic Gradient Descent (açık mavi)*
3) *ADAM (sarı)*

Kaynak Kodu

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <ctype.h>
#include <math.h>
#include <windows.h>
#include <conio.h>
#include <time.h>
#define ITER 1000
#define ITER2 10000
#define ITER3 500
#define MAX_WORD_SIZE 250

typedef struct // Dictionary struct variable.
{
    char **dictionary;
    int wordCount;
    int size;

} wordList;

typedef struct // hotVectors struct variable.
{
    double **vectorList;
    int vectorCount;

} hotVectors;

bool isWordInList(wordList *wordList, char *word);
void fillWordList(wordList *wordList, char *filename);
void trimPunctuation(char *word);
void createVector(wordList *wordList, char *sentence, hotVectors *vectors, int
sentenceIndex);
void findSentences(wordList *wordList, hotVectors *vectors, char *filename);
bool isWordInList(wordList *wordList, char *word);
void fillWordList(wordList *wordList, char *filename);
void allocateList(wordList *list);
void allocateVectors(hotVectors *vectors, int wordCount, int vectorCount);
void printDictionary(wordList *list);
void createRealData(double *realY);
void createTestRealData(double *testData);
void gradientDescent(wordList *list, double *realData, hotVectors *vector, double *w);
void ADAM(wordList *list, double *realData, hotVectors *vector, double *w);
void stochasticGradientDescent(wordList *list, double *realData, hotVectors *vector, double
*w);
void calculateResultFromMethods(double *w, hotVectors *testData, double *realData, int
wordCount);
void assignParameters(double *parameters, double assignValue, int len);
void trainMethods(wordList *dictionary, double *realData, hotVectors *vectors, double
***initialParameters, double *initialValue);
void writeParameters(char *filename, double ***initialParameters, int wordCount);
void writeErrors(double errors[], int iter);
```

```c
int main()
{
    //  DECLARATIONS & ALLOCATIONS //
    srand(time(NULL));
    int i, j;
    char *classA = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    char *classB = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    char *test = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    char *trainingData = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    char *excelfile = "wparametersfinal.csv";
    wordList *dictionary = (wordList *)malloc(sizeof(wordList));
    hotVectors *vectors = (hotVectors *)malloc(sizeof(hotVectors));
    hotVectors *vectorsTest = (hotVectors *)malloc(sizeof(hotVectors));
    allocateList(dictionary);
    //  DECLARATIONS & ALLOCATIONS //

    // GETTING THE FILE NAMES //
    printf("\n# Give the dictionary data for class A : ");
    scanf("%s", classA);
    printf("\nEntered file name : %s", classA);
    printf("\n# Give the dictionary data for class B : ");
    scanf("%s", classB);
    printf("\nEntered file name : %s", classB);
    printf("\n# Give the training data we will create hot vector : ");
    scanf("%s", trainingData);
    printf("\nEntered file name : %s", trainingData);
    printf("\n# Give the test data : ");
    scanf("%s", test);
    printf("\nEntered file name : %s", test);
    // system("cls");
    //  GETTING THE FILE NAMES //

    // CREATING THE DICTIONARY FROM CLASS A AND CLASS B //
    fillWordList(dictionary, classA);
    // system("cls");
    fillWordList(dictionary, classB);
    // system("cls");
    //  CREATING THE DICTIONARY FROM CLASS A AND CLASS B //

    // ALLOCATING VECTORS
    allocateVectors(vectors, dictionary->wordCount, MAX_WORD_SIZE);
    allocateVectors(vectorsTest, dictionary->wordCount, MAX_WORD_SIZE);
    // ALLOCATING VECTORS

    // CREATING HOT VECTORS FROM TRAINING DATA
    findSentences(dictionary, vectors, trainingData);
    findSentences(dictionary, vectorsTest, test);
    // CREATING HOT VECTORS FROM TRAINING DATA

    // ASSIGNING INITIAL PARAMETERS
    double ***initialParameters = (double ***)calloc(3, sizeof(double **));
    double *initialValue = (double *)calloc(5, sizeof(double));

    initialValue[0] = 0.0;
    initialValue[1] = 0.02;
    initialValue[2] = -0.02;
    initialValue[3] = 0.1;
    initialValue[4] = -0.3;
    for (j = 0; j < 3; j++)
    {
        initialParameters[j] = (double **)calloc(5, sizeof(double **));
        for (i = 0; i < 5; i++)
        {
            initialParameters[j][i] = (double *)calloc(dictionary->wordCount,
sizeof(double));
            assignParameters(initialParameters[j][i], initialValue[i], dictionary-
>wordCount);
        }
    }
    // ASSIGNING INITIAL PARAMETERS

    // ASSIGNING REAL AND TEST DATA FOR DEVELOPMENT
    double *realData = (double *)calloc(160, sizeof(double));
    double *testData = (double *)calloc(40, sizeof(double));
    createRealData(realData);
    createTestRealData(testData);
    // ASSIGNING REAL AND TEST DATA FOR DEVELOPMENT

    // TRAINING ALL METHODS FOR DIFFERENT INITIAL VALUES
    trainMethods(dictionary, realData, vectors, initialParameters, initialValue);
    // TRAINING ALL METHODS FOR DIFFERENT INITIAL VALUES
    // Sleep(1000);

    // TESTING ALL METHODS WITH TEST DATA
    for (i = 0; i < 5; i++)
    {
        calculateResultFromMethods(initialParameters[0][i], vectorsTest, testData,
dictionary->wordCount);
        printf("\nGRADIENT DESCENT -> Given Initial Value : %lf", initialValue[i]);
        Sleep(5000);
        calculateResultFromMethods(initialParameters[1][i], vectorsTest, testData,
dictionary->wordCount);
        printf("\nSTOHASTIC GRADIENT DESCENT -> Given Initial Value : %lf",
initialValue[i]);
        Sleep(5000);
        calculateResultFromMethods(initialParameters[2][i], vectorsTest, testData,
dictionary->wordCount);
        printf("\nADAM -> Given Initial Value : %lf", initialValue[i]);
        Sleep(5000);
    }
    // TESTING ALL METHODS WITH TEST DATA

    // WRITES FINAL PARAMETERS INTO A FILE
    writeParameters(excelfile, initialParameters, dictionary->wordCount);
    // WRITES FINAL PARAMETERS INTO A FILE

    return 0;
    //  END OF THE CODE
}
double dtanh(double x)
{
    double coshx = cosh(x);
    return 1.0 / (coshx * coshx);
}
void trimPunctuation(char *word)
```

```c
{
    int i = 0;
    while (i < strlen(word))
    {
        while (word[i] != '\0' && !ispunct(word[i]) && word[i] != '\n')
        {
            i++;
        }
        int j = i;
        // Trim trailing punctuation
        while (j < strlen(word))
        {
            word[j] = word[j + 1];
            j++;
        }
        word[j + 1] = '\0';
    }
    word[i + 1] = '\0';
}

void createVector(wordList *wordList, char *sentence, hotVectors *vectors, int
sentenceIndex)
{
    int head, tail, j, i;
    head = 0;
    tail = 0;
    char buffer[500];
    while (head < strlen(sentence))
    {
        j = 0;
        while (isalpha(sentence[tail]))
        {
            buffer[j] = sentence[tail];
            tail++;
            j++;
        }
        buffer[j] = '\0';
        for (i = 0; i < wordList->wordCount; i++)
        {
            if (!strcmp(buffer, wordList->dictionary[i]))
            {
                vectors->vectorList[sentenceIndex][i] = 1;
                break;
            }
        }
        tail++;
        head = tail;
    }
}

void findSentences(wordList *wordList, hotVectors *vectors, char *filename)
{
    int sentenceCount = 0;
    FILE *file = fopen(filename, "r");
    if (file == NULL)
    {
        perror("Error opening file");
        return;
    }
    char sentence[500];
    int c;
    int sentenceIndex = 0;
    while ((c = fgetc(file)) != EOF)
    {
        sentence[sentenceIndex++] = c;
        // Checking if we've reached the end of a sentence
        if (c == '.' || c == '?' || c == '!')
        {
            c = fgetc(file);
            if (!isdigit(c))
            {
                sentence[sentenceIndex++] = '\0';
                trimPunctuation(sentence);
                strlwr(sentence);
                createVector(wordList, sentence, vectors, sentenceCount); // Creating
vector if a sentence is found
                sentenceCount = sentenceCount + 1;
                sentenceIndex = 0;
                memset(sentence, 0, sizeof(sentence));
            }
            sentence[sentenceIndex++] = c;
        }
    }

    fclose(file);
    vectors->vectorCount = sentenceCount;
    printf("\n%d vectors are created from file %s. Process completed.", vectors-
>vectorCount, filename);
}

bool isWordInList(wordList *wordList, char *word) // Checks if the word is in list.
{
    int i = 0;
    while (i < wordList->wordCount && strcmp(word, wordList->dictionary[i]))
    {
        i++;
    }
    if (i < wordList->wordCount)
    {
        return true;
    }
    else
        return false;
}

void fillWordList(wordList *wordList, char *filename) // Reading a file and putting words
to dictionary
{
    FILE *file = fopen(filename, "r");
    if (file == NULL)
    {
        perror("\n!ERROR! File cannot read!");
        return;
    }
    printf("\n%s is reading, please wait...", filename);
    char *currentWord = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    while (fscanf(file, "%s", currentWord) != EOF)
    {
        trimPunctuation(currentWord);
        currentWord = strlwr(currentWord);
        if (!isdigit(currentWord[0]) && !isWordInList(wordList, currentWord))
        {
            if (wordList->wordCount + 1 > wordList->size)
            {
                wordList->size += 100;
                wordList->dictionary = realloc(wordList->dictionary, (wordList->size) *
sizeof(char *));
                int i;
                for (i = wordList->wordCount; i < wordList->wordCount + 100; i++)
                    wordList->dictionary[i] = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
            }
            strcpy(wordList->dictionary[wordList->wordCount], currentWord);
            wordList->wordCount++;
        }
    }
    fclose(file);
    printf("\nWords added from %s. File closed successfully.", filename);
}

void allocateList(wordList *list) // Allocating memory for dictionary
{
    int i, j;
    list->wordCount = 0;
    list->size = MAX_WORD_SIZE;
    list->dictionary = (char **)malloc(sizeof(char *) * MAX_WORD_SIZE);
    for (i = 0; i < MAX_WORD_SIZE; i++)
    {
        list->dictionary[i] = (char *)malloc(sizeof(char) * MAX_WORD_SIZE);
    }
}
void allocateVectors(hotVectors *vectors, int wordCount, int vectorCount) // Allocating
memory for hot vectorss
{
    int i, j;
    vectors->vectorList = (double **)malloc(sizeof(double *) * vectorCount);
    for (i = 0; i < vectorCount; i++)
    {
        vectors->vectorList[i] = (double *)malloc(sizeof(double) * wordCount);
    }
}
void printDictionary(wordList *list)
{
    system("cls");
    printf("\n$$$ DICTIONARY $$$\n");
    int i;
    for (i = 0; i < list->wordCount; i++)
    {
        printf("\n-> Word %d : %s", i + 1, list->dictionary[i]);
    }
}

void createHotVector(wordList *list, double *vector)
{
    char *currentText = (char *)calloc(10000, sizeof(char));
    int i;
    for (i = 0; i < list->wordCount; i++)
    {
    }
}

void createRealData(double *realY)
{
    int i;
    for (i = 0; i < 160; i++)
    {
        if (i < 80)
            realY[i] = 1;
        else
            realY[i] = -1;
    }
    printf("\nReal data has been created.");
}
void createTestRealData(double *testData)
{
    int i;
    for (i = 0; i < 40; i++)
    {
        if (i < 20)
            testData[i] = 1.0;
        else
            testData[i] = -1.0;
    }
    printf("\nTest data has been created.");
}

void gradientDescent(wordList *list, double *realData, hotVectors *vector, double *w) //
Gradient Descent Algorithm
{
    int i, j, k;
    double LR = 0.03;
    double grad, wx = 0;
    double errors[ITER];
    double error = 0;
    clock_t start = clock();
    for (i = 0; i < ITER; i++)
    {
        for (j = 0; j < vector->vectorCount; j++) // Sentence count
        {
            wx = 0;
            for (k = 0; k < list->wordCount; k++) // Word count
            {
                wx += w[k] * vector->vectorList[j][k];
            }

            grad = ((realData[j] - tanh(wx)));
            for (k = 0; k < list->wordCount; k++)
            {
                w[k] = w[k] - ((LR * grad * dtanh(wx) * vector->vectorList[j][k]) * (-2.0 /
list->wordCount)); // Gradient Descent implementation
            }
            error += grad * grad;
        }
        errors[i] = error / vector->vectorCount; // Calculating errors
        error = 0;
    }
    clock_t end = clock();
    double execTime = (double)(end - start) / CLOCKS_PER_SEC; // Calculating time
    printf("\nExecution time GD: %lf", execTime);
}

void stochasticGradientDescent(wordList *list, double *realData, hotVectors *vector, double
*w)
{
    int i, j, k, l;
    double LR = 0.3;
    double grad, wx = 0;
    double errors[ITER2];
    double error = 0;
    clock_t start = clock();
    for (i = 0; i < ITER2; i++)
    {
        j = rand() % ((0 - 160 + 1) + 0);
        wx = 0.0;
        for (k = 0; k < list->wordCount; k++)
        {
            wx += w[k] * vector->vectorList[j][k];
        }
        grad = ((realData[j] - tanh(wx)));
        for (k = 0; k < list->wordCount; k++)
        {
            w[k] = w[k] - ((LR * grad * dtanh(wx) * vector->vectorList[j][k]) * (-2.0 /
list->wordCount));
        }
        error += grad * grad;
        errors[i] = error / vector->vectorCount;
        error = 0;
    }
    clock_t end = clock();
    double execTime = (double)(end - start) / CLOCKS_PER_SEC;
    printf("\nExecution time for SGD : %lf", execTime);
}

void ADAM(wordList *list, double *realData, hotVectors *vector, double *w)
{
    int i, j, k;
    double b = 0, vw = 0, vb = 0, sw = 0, sb = 0;
    double beta1 = 0.9, beta2 = 0.999, epsilon = 0.00000001;
    double grad = 1, y_hat = 0, diff;
    double wx;
    double errors[ITER3];
    double error = 0;
    clock_t start = clock();
    for (i = 0; i < ITER3; i++)
    {
        for (j = 0; j < vector->vectorCount; j++)
        {
            y_hat = 0;
            wx = 0;
            grad = 0;
```

```c
            for (k = 0; k < list->wordCount; k++)
            {
                y_hat += w[k] * vector->vectorList[j][k];
            }
            wx = tanh(y_hat);
            diff = (wx - realData[j]);
            grad = 0;
            for (k = 0; k < list->wordCount; k++)
            {
                grad = (2.0 / list->wordCount) * (diff) * (dtanh(y_hat)) * (vector-
>vectorList[j][k]);
                sw = sw * beta2 + (1 - beta2) * grad * grad;
                vw = vw * beta1 + (1 - beta1) * grad;
                w[k] = w[k] - ((0.001 * vw) / (sqrt(sw + epsilon)));
            }
            error += diff * diff;
        }
        errors[i] = error / vector->vectorCount;
        error = 0;
    }
    clock_t end = clock();
    double execTime = (double)(end - start) / CLOCKS_PER_SEC;
    printf("\nExecution time for ADAM : %lf", execTime);
}

void calculateResultFromMethods(double *w, hotVectors *testData, double *realData, int
wordCount)
{
    int i, j, trueCount = 0, falseCount = 0;
    for (i = 0; i < testData->vectorCount; i++)
    {
        double result = 0.0;
        for (j = 0; j < wordCount; j++)
        {
            result += w[j] * testData->vectorList[i][j];
        }
        result = tanh(result);
        printf("\nReal value %.10lf | %.10lf Estimated value", realData[i], result);

        if (pow(realData[i] - result, 2) < 0.75)
        {
            trueCount++;
            printf("\n%dth sentence is estimated correctly!", i + 1);
            /*
            if (realData[i] < 0)
                printf("\nThis sentence belongs to Movie class.", i + 1);
            else
                printf("\nThis sentence belongs to Technology class. ", i + 1);
                */
        }
        else
        {
            // printf("\n%dth sentence is estimated incorrectly!", i + 1);
            falseCount++;
        }
    }
    printf("\nTrue count %d | %d False count", trueCount, falseCount);
}
```

```c
void assignParameters(double *parameters, double assignValue, int len)
{
    int i;
    printf("\nAssigning value : %lf", assignValue);
    for (i = 0; i < len; i++)
        parameters[i] = assignValue;
}
void trainMethods(wordList *dictionary, double *realData, hotVectors *vectors, double
***initialParameters, double *initialValue)
{
    int i = 0, j = 0;
    for (i = 0; i < 5; i++)
    {
        printf("\nGiven Initial Value : %lf", initialValue[i]);
        gradientDescent(dictionary, realData, vectors, initialParameters[0][i]);
        stochasticGradientDescent(dictionary, realData, vectors, initialParameters[1][i]);
        ADAM(dictionary, realData, vectors, initialParameters[2][i]);
    }
}
void writeParameters(char *filename, double ***initialParameters, int wordCount)
{
    int i, j, k;
    FILE *fptr = fopen(filename, "w");
    if (!fptr)
    {
        perror("\nError! Cannot open file!");
        return;
    }

    for (i = 0; i < wordCount; i++)
    {
        fprintf(fptr, "%lf,", initialParameters[0][0][i]);
        fprintf(fptr, "%lf,", initialParameters[0][1][i]);
        fprintf(fptr, "%lf,", initialParameters[0][2][i]);
        fprintf(fptr, "%lf,", initialParameters[0][3][i]);
        fprintf(fptr, "%lf,", initialParameters[0][4][i]);
        fprintf(fptr, "%s,", "1");

        fprintf(fptr, "\n%s", "");
    }
    for (i = 0; i < wordCount; i++)
    {
        fprintf(fptr, "%lf,", initialParameters[1][0][i]);
        fprintf(fptr, "%lf,", initialParameters[1][1][i]);
        fprintf(fptr, "%lf,", initialParameters[1][2][i]);
        fprintf(fptr, "%lf,", initialParameters[1][3][i]);
        fprintf(fptr, "%lf,", initialParameters[1][4][i]);
        fprintf(fptr, "%s,", "2");

        fprintf(fptr, "\n%s", "");
    }
    for (i = 0; i < wordCount; i++)
    {
        fprintf(fptr, "%lf,", initialParameters[2][0][i]);
        fprintf(fptr, "%lf,", initialParameters[2][1][i]);
        fprintf(fptr, "%lf,", initialParameters[2][2][i]);
        fprintf(fptr, "%lf,", initialParameters[2][3][i]);
        fprintf(fptr, "%lf,", initialParameters[2][4][i]);
        fprintf(fptr, "%s,", "3");

        fprintf(fptr, "\n%s", "");
    }
    printf("\nDatas are written in %s successfully.", filename);
    fclose(fptr);
}
void writeErrors(double errors[], int iter)
{
    int i, j;
    FILE *fptr = fopen("errorGD.csv", "w");
    if (!fptr)
    {
        perror("\nError! Cannot open file!");
        return;
    }
    fprintf(fptr, "%s\n", "");
    for (i = 0; i < iter; i++)
    {
        fprintf(fptr, "%lf,", errors[i]);
        fprintf(fptr, "%s\n", "");
    }
    fprintf(fptr, "%s\n", "ITEREND");
    fclose(fptr);
    printf("\nErrors has been written successfully.");
}
```

Veri Setleri

“Teknoloji” sınıfı sözlük veri seti

1. Technology has revolutionized communication, making it faster and more efficient.

2. Smartphones have become ubiquitous, providing access to a world of information at our fingertips.

3. The internet has connected people globally, fostering a digital community.

4. Artificial intelligence is transforming industries, automating tasks and enhancing productivity.

5. Virtual reality offers immersive experiences, from gaming to training simulations.

6. Cloud computing enables the storage and access of data from anywhere in the world.

7. Wearable technology, like smartwatches, tracks health metrics and enhances daily life.

8. 3D printing has revolutionized manufacturing, allowing for rapid prototyping and customization.

9. Nanotechnology explores materials and devices at the nanoscale, promising groundbreaking advancements.

10. Autonomous vehicles are a frontier in transportation, aiming to redefine the way we travel.

11. Biotechnology is advancing medical treatments and genetic research.

12. Quantum computing holds the potential to solve complex problems at unprecedented speeds.

13. Augmented reality overlays digital information onto the real world, enhancing perception.

14. Robotics is reshaping industries, from manufacturing to healthcare and beyond.

15. The Internet of Things connects everyday devices, creating a network of smart, interconnected systems.

16. Cybersecurity measures protect against online threats, ensuring the integrity of digital systems.

17. E-commerce has transformed the way we shop, with online transactions becoming the norm.

18. Renewable energy technologies, such as solar and wind power, are crucial for a sustainable future.

19. Big data analytics harness large datasets to derive valuable insights and inform decision-making.

20. 5G technology is ushering in faster and more reliable wireless communication.

21. Machine learning algorithms improve over time, optimizing processes and predictions.

22. E-learning platforms provide accessible education opportunities globally.

23. GPS technology has revolutionized navigation, making maps and directions readily available.

24. Biometrics, like fingerprint and facial recognition, enhance security and authentication.

25. Quantum cryptography promises unbreakable encryption for secure communication.

26. Drones are used for various applications, from aerial photography to delivery services.

27. Smart homes integrate technology for automated and energy-efficient living spaces.

28. Blockchain technology ensures secure and transparent transactions in decentralized systems.

29. Holographic displays offer a futuristic way to visualize information in three dimensions.

30. Gene editing technologies, like CRISPR, hold the potential to treat genetic diseases.

31. Voice-activated assistants, such as Siri and Alexa, respond to verbal commands for convenience.

32. Cryptocurrencies, like Bitcoin, are digital currencies operating on decentralized blockchain technology.

33. 3D scanners capture detailed spatial information for various applications, from design to archaeology.

34. Satellite technology enables global communication, weather monitoring, and navigation.

35. Quantum sensors can detect minute changes in physical quantities, with applications in various fields.

36. Smart cities use technology to optimize urban infrastructure for efficiency and sustainability.

37. Mobile payment systems facilitate cashless transactions using smartphones.

38. Augmented reality in healthcare aids surgeons with real-time information during procedures.

39. Responsive web design ensures seamless user experiences across various devices.

40. Precision agriculture utilizes technology to optimize farming practices and maximize yields.

41. Neural interfaces connect technology directly to the human brain, offering potential for medical breakthroughs.

42. 4K and 8K resolution in displays provide stunning visual clarity for entertainment and professional use.

43. Smart fabrics integrate technology into clothing for enhanced functionality, such as temperature regulation.

44. Quantum sensors can detect gravitational waves, opening new frontiers in astrophysics.

45. Electric vehicles are becoming more prevalent, reducing dependence on traditional fossil fuels.

46. Genetic sequencing technologies enable the mapping of individual genomes for personalized medicine.

47. E-waste management addresses the environmental impact of discarded electronic devices.

48. The development of exoskeletons enhances mobility for individuals with mobility impairments.

49. Quantum communication promises secure transmission of information using quantum principles.

50. Drone technology is used in agriculture for crop monitoring and precision spraying.

51. Smart agriculture employs sensors and data analytics to optimize crop management.

52. Advanced materials, such as graphene, offer unique properties for various technological applications.

53. Telemedicine leverages technology for remote healthcare consultations and monitoring.

54. Internet censorship technologies restrict access to certain online content in some regions.

55. Mobile health apps enable individuals to monitor and manage their well-being on the go.

56. Biometric wearables, like fitness trackers, measure and analyze physiological data.

57. Robotics in healthcare assists with surgeries, rehabilitation, and patient care.

58. 3D bioprinting creates living tissue structures for medical and research purposes.

59. Satellite imaging aids in environmental monitoring, disaster response, and urban planning.

60. Autonomous drones are employed for surveillance, search and rescue, and environmental monitoring.

61. Edge computing processes data closer to the source, reducing latency for real-time applications.

62. Quantum sensors can be used for early detection of diseases through biomarker analysis.

63. Smart glasses offer augmented reality experiences for various professional and personal applications.

64. Internet censorship circumvention tools enable access to restricted content in some regions.

65. The rise of digital currencies is reshaping traditional notions of banking and finance.

66. Quantum computing algorithms have the potential to revolutionize optimization problems.

67. Neural networks in artificial intelligence mimic the human brain's learning and decision-making processes.

68. Electric planes are being developed as a sustainable alternative to traditional aviation.

69. Biodegradable electronics address environmental concerns related to electronic waste.

70. Gene therapy uses genetic engineering to treat or prevent diseases at the molecular level.

71. Telepresence technology allows individuals to remotely participate in events or meetings.

72. Quantum sensors in environmental monitoring track changes in air and water quality.

73. The dark web utilizes encrypted networks for anonymous online activities.

74. E-sports, competitive video gaming, has become a major industry with a global audience.

75. Adaptive learning platforms customize educational content based on individual progress and needs.

76. Gesture recognition technology allows users to interact with devices through hand movements.

77. Cyber-physical systems integrate computational algorithms with physical processes for improved efficiency.

78. Quantum computing has the potential to revolutionize cryptography by breaking current encryption methods.

79. Biometric authentication methods, like iris scans, enhance security in various applications.

80. Hyperloop technology aims to revolutionize transportation with high-speed capsule travel in low-pressure tubes.

81. Brain-machine interfaces enable direct communication between the brain and external devices.

82. Quantum sensors in agriculture monitor soil conditions and crop health for optimal yield.

83. Smart grid technology enhances the efficiency and reliability of electrical power distribution.

84. Predictive analytics use data and algorithms to forecast future trends and outcomes.

85. Quantum computing's parallel processing capabilities can solve complex problems faster than classical computers.

86. Virtual classrooms and online learning platforms have become integral to modern education.

87. The development of lab-grown meat addresses environmental and ethical concerns in traditional agriculture.

88. Quantum sensors in navigation systems provide highly accurate positioning information.

89. Cognitive computing systems simulate human thought processes, aiding in complex decision-making.

90. Biometric identification is used in border control and immigration for enhanced security.

91. Quantum communication networks ensure secure and private communication using quantum principles.

92. Swarm robotics involves the coordination of multiple robots to accomplish tasks collaboratively.

93. The use of blockchain technology in supply chain management enhances transparency and traceability.

94. Quantum sensors in weather monitoring provide precise data for forecasting and research.

95. Haptic technology simulates the sense of touch, enhancing virtual and augmented reality experiences.

96. The development of quantum computers has the potential to revolutionize drug discovery by simulating complex molecular interactions.

97. Robotic exosuits are being explored to enhance the physical capabilities of soldiers and first responders.

98. Edge AI processing enables smart devices to make rapid decisions locally, reducing dependence on cloud servers.

99. Quantum key distribution ensures secure communication channels by leveraging the principles of quantum mechanics.

100. Hyperautomation integrates advanced technologies, such as AI and robotic process automation, to streamline and optimize complex business processes.

“Film” sınıfı sözlük veri seti

1. Movies transport audiences to fantastical worlds.
2. Film directors shape narratives through visual storytelling.
3. Cinematography captures the essence of a story.
4. Movie genres range from action to romance.
5. Blockbusters attract large audiences with thrilling spectacles.
6. Classic films stand the test of time.
7. Sequels continue beloved stories.
8. Film festivals showcase diverse cinematic talents.
9. Actors bring characters to life on the big screen.
10. Movie soundtracks enhance emotional impact.
11. Special effects create stunning visual experiences.
12. Hollywood produces numerous iconic films.
13. Indie films explore unique narratives.
14. Documentaries shed light on real-life stories.
15. Movie theaters offer communal viewing experiences.
16. Streaming platforms provide convenient access.
17. Directors use symbolism to convey deeper meanings.
18. Film critics analyze performances and storytelling.
19. Moviegoers enjoy the escapism of cinema.
20. Film awards recognize outstanding achievements.
21. Cult films develop passionate fan bases.
22. Movie posters serve as visual teasers.
23. Cinematic universes connect multiple storylines.
24. Animated films appeal to audiences of all ages.
25. Foreign films offer diverse cultural perspectives.
26. Film festivals celebrate artistic creativity.
27. Popcorn is a classic movie-watching snack.
28. Movie quotes become part of popular culture.
29. 3D technology adds depth to visual experiences.
30. Film noir explores dark and mysterious themes.
31. Nostalgic movies evoke sentimental feelings.
32. The silver screen represents a timeless medium.
33. Movie premieres attract star-studded crowds.
34. Character development is crucial to engaging plots.
35. Movie franchises span multiple installments.
36. Screenplays provide the foundation for storytelling.
37. Opening credits set the tone for the film.
38. Movie merchandising capitalizes on fan enthusiasm.
39. Remakes offer fresh perspectives on classic stories.
40. Directors collaborate with cinematographers for visual impact.
41. Suspenseful films keep audiences on the edge of their seats.
42. Movie scores evoke powerful emotions.

43. Horror films play on primal fears.

44. Moviegoers debate plot twists and endings.

45. Film adaptations bring books to life.

46. Movie magic is created through skilled editing.

47. The red carpet is synonymous with glamorous film events.

48. Auteur directors imprint their unique style on films.

49. Film studios invest in blockbuster franchises.

50. Movies explore complex moral and ethical dilemmas.

51. Action sequences showcase choreographed stunts.

52. Costume design enhances character authenticity.

53. Cinematic experiences vary from IMAX to indie theaters.

54. Film ratings guide viewers on age-appropriate content.

55. Biopics chronicle the lives of real individuals.

56. Movie reviews influence audience expectations.

57. CGI transforms imaginary worlds into reality.

58. Film societies promote appreciation for the art form.

59. Movie releases generate anticipation and excitement.

60. Film preservation ensures classics endure for future generations.

61. Movies reflect societal values and concerns.

62. The film industry has a global impact on culture.

63. Movie marathons are a popular pastime.

64. Famous movie quotes become cultural catchphrases.

65. The Cannes Film Festival is a prestigious industry event.

66. Plot twists keep audiences guessing until the end.

67. Film historians study the evolution of cinema.

68. Movie buffs collect memorabilia from their favorite films.

69. Movie sets are carefully designed for authenticity.

70. Directors collaborate with production designers for visual aesthetics.

71. Cinematic achievements are celebrated at award ceremonies.

72. Movie characters resonate with audiences on a personal level.

73. Sound design enhances the immersive experience.

74. Films explore the human condition and emotions.

75. Hollywood legends leave a lasting impact on the industry.

76. Movie studios compete for box office supremacy.

77. Moviegoers appreciate compelling storytelling.

78. Film trailers generate buzz and anticipation.

79. Movie posters often feature iconic imagery.

80. Moviegoers debate the merits of practical effects versus CGI.

81. Film studios invest in cutting-edge technology for production.

82. Movie adaptations of video games bring virtual worlds to life.

83. Film education programs nurture future filmmakers.

84. Directors use symbolism to convey deeper meanings.

85. Movie franchises expand into television series.

86. Classic film stars are remembered for their timeless performances.

87. Film ratings provide guidance on content suitability.

88. Movie makeup transforms actors into characters.

89. Filmmaking is a collaborative and interdisciplinary art.

90. Movies inspire fashion trends and iconic styles.

91. The Academy Awards honor excellence in cinematic achievements.

92. Movie enthusiasts attend film festivals to discover hidden gems.

93. Genre crossovers offer refreshing storytelling twists.

94. Animated shorts showcase creativity in a condensed format.

95. Movie premieres generate excitement and media coverage.

96. Classic film scores become ingrained in popular culture.

97. Movie production involves meticulous planning and coordination.

98. Film trivia enhances the viewing experience for enthusiasts.

99. Moviegoers appreciate thought-provoking and original concepts.

100. Cinema has the power to evoke a wide range of emotions.

Eğitim veri seti

Technology has revolutionized communication, making it faster and more efficient.

Smartphones have become ubiquitous, providing access to a world of information at our fingertips.

The internet has connected people globally, fostering a digital community.

Artificial intelligence is transforming industries, automating tasks and enhancing productivity.

Virtual reality offers immersive experiences, from gaming to training simulations.

Cloud computing enables the storage and access of data from anywhere in the world.

Wearable technology, like smartwatches, tracks health metrics and enhances daily life.

3D printing has revolutionized manufacturing, allowing for rapid prototyping and customization.

Biotechnology is advancing medical treatments and genetic research.

Quantum computing holds the potential to solve complex problems at unprecedented speeds.

Augmented reality overlays digital information onto the real world, enhancing perception.

Robotics is reshaping industries, from manufacturing to healthcare and beyond.

The Internet of Things connects everyday devices, creating a network of smart, interconnected systems.

Cybersecurity measures protect against online threats, ensuring the integrity of digital systems.

E-commerce has transformed the way we shop, with online transactions becoming the norm.

Renewable energy technologies, such as solar and wind power, are crucial for a sustainable future.

Machine learning algorithms improve over time, optimizing processes and predictions.

E-learning platforms provide accessible education opportunities globally.

GPS technology has revolutionized navigation, making maps and directions readily available.

Biometrics, like fingerprint and facial recognition, enhance security and authentication.

Quantum cryptography promises unbreakable encryption for secure communication.

Drones are used for various applications, from aerial photography to delivery services.

Smart homes integrate technology for automated and energy-efficient living spaces.

Blockchain technology ensures secure and transparent transactions in decentralized systems.

Voice-activated assistants, such as Siri and Alexa, respond to verbal commands for convenience.

Cryptocurrencies, like Bitcoin, are digital currencies operating on decentralized blockchain technology.

3D scanners capture detailed spatial information for various applications, from design to archaeology.

Satellite technology enables global communication, weather monitoring, and navigation.

Quantum sensors can detect minute changes in physical quantities, with applications in various fields.

Smart cities use technology to optimize urban infrastructure for efficiency and sustainability.

Mobile payment systems facilitate cashless transactions using smartphones.

Augmented reality in healthcare aids surgeons with real-time information during procedures.

Neural interfaces connect technology directly to the human brain, offering potential for medical breakthroughs.

4K and 8K resolution in displays provide stunning visual clarity for entertainment and professional use.

Smart fabrics integrate technology into clothing for enhanced functionality, such as temperature regulation.

Quantum sensors can detect gravitational waves, opening new frontiers in astrophysics.

Electric vehicles are becoming more prevalent, reducing dependence on traditional fossil fuels.

Genetic sequencing technologies enable the mapping of individual genomes for personalized medicine.

E-waste management addresses the environmental impact of discarded electronic devices.

The development of exoskeletons enhances mobility for individuals with mobility impairments.

Smart agriculture employs sensors and data analytics to optimize crop management.

Advanced materials, such as graphene, offer unique properties for various technological applications.

Telemedicine leverages technology for remote healthcare consultations and monitoring.

Internet censorship technologies restrict access to certain online content in some regions.

Mobile health apps enable individuals to monitor and manage their well-being on the go.

Biometric wearables, like fitness trackers, measure and analyze physiological data.

Robotics in healthcare assists with surgeries, rehabilitation, and patient care.

3D bioprinting creates living tissue structures for medical and research purposes.

Edge computing processes data closer to the source, reducing latency for real-time applications.

Quantum sensors can be used for early detection of diseases through biomarker analysis.

Smart glasses offer augmented reality experiences for various professional and personal applications.

Internet censorship circumvention tools enable access to restricted content in some regions.

The rise of digital currencies is reshaping traditional notions of banking and finance.

Quantum computing algorithms have the potential to revolutionize optimization problems.

Neural networks in artificial intelligence mimic the human brain's learning and decision-making processes.

Electric planes are being developed as a sustainable alternative to traditional aviation.

Telepresence technology allows individuals to remotely participate in events or meetings.

Quantum sensors in environmental monitoring track changes in air and water quality.

The dark web utilizes encrypted networks for anonymous online activities.

E-sports, competitive video gaming, has become a major industry with a global audience.

Adaptive learning platforms customize educational content based on individual progress and needs.

Gesture recognition technology allows users to interact with devices through hand movements.

Cyber-physical systems integrate computational algorithms with physical processes for improved efficiency.

Quantum computing has the potential to revolutionize cryptography by breaking current encryption methods.

Brain-machine interfaces enable direct communication between the brain and external devices.

Quantum sensors in agriculture monitor soil conditions and crop health for optimal yield.

Smart grid technology enhances the efficiency and reliability of electrical power distribution.

Predictive analytics use data and algorithms to forecast future trends and outcomes.

Quantum computing's parallel processing capabilities can solve complex problems faster than classical computers.

Virtual classrooms and online learning platforms have become integral to modern education.

The development of lab-grown meat addresses environmental and ethical concerns in traditional agriculture.

Quantum sensors in navigation systems provide highly accurate positioning information.

Quantum communication networks ensure secure and private communication using quantum principles.

Swarm robotics involves the coordination of multiple robots to accomplish tasks collaboratively.

The use of blockchain technology in supply chain management enhances transparency and traceability.

Quantum sensors in weather monitoring provide precise data for forecasting and research.

Haptic technology simulates the sense of touch, enhancing virtual and augmented reality experiences.

The development of quantum computers has the potential to revolutionize drug discovery by simulating complex molecular interactions.

Robotic exosuits are being explored to enhance the physical capabilities of soldiers and first responders.

Edge AI processing enables smart devices to make rapid decisions locally, reducing dependence on cloud servers.

Movie transport audiences to fantastical worlds.

Film directors shape narratives through visual storytelling.

Cinematography captures the essence of a story.

Movie genres range from action to romance.

Blockbusters attract large audiences with thrilling spectacles.

Classic films stand the test of time.

Sequels continue beloved stories.

Film festivals showcase diverse cinematic talents.

Special effects create stunning visual experiences.

Hollywood produces numerous iconic films.

Indie films explore unique narratives.

Documentaries shed light on real-life stories.

Movie theaters offer communal viewing experiences.

Streaming platforms provide convenient access.

Directors use symbolism to convey deeper meanings.

Film critics analyze performances and storytelling.

Cult films develop passionate fan bases.

Movie posters serve as visual teasers.

Cinematic universes connect multiple storylines.

Animated films appeal to audiences of all ages.

Foreign films offer diverse cultural perspectives.

Film festivals celebrate artistic creativity.

Popcorn is a classic movie-watching snack.

Movie quotes become part of popular culture.

Nostalgic movie evoke sentimental feelings.

The silver screen represents a timeless medium.

Movie premieres attract star-studded crowds.

Character development is crucial to engaging plots.

Movie franchises span multiple installments.

Screenplays provide the foundation for storytelling.

Opening credits set the tone for the film.

Movie merchandising capitalizes on fan enthusiasm.

Suspenseful films keep audiences on the edge of their seats.

Movie scores evoke powerful emotions.

Horror films play on primal fears.

Moviegoers debate plot twists and endings.

Film adaptations bring books to life.

Movie magic is created through skilled editing.

The red carpet is synonymous with glamorous film events.

Auteur directors imprint their unique style on films.

Action sequences showcase choreographed stunts.

Costume design enhances character authenticity.

Cinematic experiences vary from IMAX to indie theaters.

Film ratings guide viewers on age-appropriate content.

Biopics chronicle the lives of real individuals.

Movie reviews influence audience expectations.

CGI transforms imaginary worlds into reality.

Film societies promote appreciation for the art form.

Movie reflect societal values and concerns.

The film industry has a global impact on culture.

Movie marathons are a popular pastime.

Famous movie quotes become cultural catchphrases.

The Cannes Film Festival is a prestigious industry event.

Plot twists keep audiences guessing until the end.

Film historians study the evolution of cinema.

Movie buffs collect memorabilia from their favorite films.

Cinematic achievements are celebrated at award ceremonies.

Movie characters resonate with audiences on a personal level.

Sound design enhances the immersive experience.

Films explore the human condition and emotions.

Hollywood legends leave a lasting impact on the industry.

Movie studios compete for box office supremacy.

Moviegoers appreciate compelling storytelling.

Film trailers generate buzz and anticipation.

Film studios invest in cutting-edge technology for production.

Movie adaptations of video games bring virtual worlds to life.

Film education programs nurture future filmmakers.

Directors use symbolism to convey deeper meanings.

Movie franchises expand into television series.

Classic film stars are remembered for their timeless performances.

Film ratings provide guidance on content suitability.

Movie makeup transforms actors into characters.

The Academy Awards honor excellence in cinematic achievements.

Movie enthusiasts attend film festivals to discover hidden gems.

Genre crossovers offer refreshing storytelling twists.

Animated shorts showcase creativity in a condensed format.

Movie premieres generate excitement and media coverage.

Classic film scores become ingrained in popular culture.

Movie production involves meticulous planning and coordination.

Film trivia enhances the viewing experience for enthusiasts.

Test veri seti

Nanotechnology explores materials and devices at the nanoscale, promising groundbreaking advancements.

Autonomous vehicles are a frontier in transportation, aiming to redefine the way we travel.

Big data analytics harness large datasets to derive valuable insights and inform decision-making.

5G technology is ushering in faster and more reliable wireless communication.

Holographic displays offer a futuristic way to visualize information in three dimensions.

Gene editing technologies, like CRISPR, hold the potential to treat genetic diseases.

Responsive web design ensures seamless user experiences across various devices.

Precision agriculture utilizes technology to optimize farming practices and maximize yields.

Quantum communication promises secure transmission of information using quantum principles.

Drone technology is used in agriculture for crop monitoring and precision spraying.

Satellite imaging aids in environmental monitoring, disaster response, and urban planning.

Autonomous drones are employed for surveillance, search and rescue, and environmental monitoring.

Biodegradable electronics address environmental concerns related to electronic waste.

Gene therapy uses genetic engineering to treat or prevent diseases at the molecular level.

Biometric authentication methods, like iris scans, enhance security in various applications.

Hyperloop technology aims to revolutionize transportation with high-speed capsule travel in low-pressure tubes.

Cognitive computing systems simulate human thought processes, aiding in complex decision-making.

Biometric identification is used in border control and immigration for enhanced security.

Quantum key distribution ensures secure communication channels by leveraging the principles of quantum mechanics.

Hyperautomation integrates advanced technologies, such as AI and robotic process automation, to streamline and optimize complex business processes.

Actors bring characters to life on the big screen.

Movie soundtracks enhance emotional impact.

Moviegoers enjoy the escapism of cinema.

Film awards recognize outstanding achievements.

3D technology adds depth to visual experiences.

Film noir explores dark and mysterious themes.

Remakes offer fresh perspectives on classic stories.

Directors collaborate with cinematographers for visual impact.

Film studios invest in blockbuster franchises.

Movies explore complex moral and ethical dilemmas.

Movie releases generate anticipation and excitement.

Film preservation ensures classics endure for future generations.

Movie sets are carefully designed for authenticity.

Directors collaborate with production designers for visual aesthetics.

Movie posters often feature iconic imagery.

Moviegoers debate the merits of practical effects versus CGI.

Filmmaking is a collaborative and interdisciplinary art.

Movie inspire fashion trends and iconic styles.

Moviegoers appreciate thought-provoking and original concepts.

Cinema has the power to evoke a wide range of emotions.