# 第一章 界面类库

## 1.1 概述

实现短文本数据流分类软件的界面，表1.1给出了各类主要功能。

|  |  |
| --- | --- |
| 表1.1 界面类功能描述 | |
| 类名称 | 功能描述 |
| Main.java | 短文本数据流分类软件的主函数 |
| STSCJFrame.java | 主框架 |
| SSTSCJPanel.java | 有监督短文本数据流分类的界面实现 |
| SemiSSTSCJPanel.java | 半监督短文本数据流分类的界面实现 |

## 1.2 类SSTSCJPanel

实现有监督短文本数据流分类界面，同时检测输入参数是否正确，以下简单介绍相关成员函数：

public SSTSCJPanel(int width, int height)

——构造函数，输入参数为整个界面的宽和高。

private void initLeftJpanel()

∟private Component loadDataJpanel()

private Component extendTextJpanel()

private Component representFeatureJpanel()

private Component classifyJpanel()

——实现左边的参数设置界面，包括数据加载、文本扩展、特征表示以及分类的参数设置。

private void initRightJpanel()

——实现右边的结果输出界面。

class startTrainActionListener

——继承类ActionListener，用于响应“开始训练”按钮

∟private boolean checkFormat()

∟private Boolean isExternalCorpus(File file)

——检查输入参数格式是否正确，如果错误则向界面返回错误信息

class SSTSCThread

——继承类Thread，是响应“开始训练”按钮的线程，用于与界面实现的线程分开，使得在响应开始训练事件时仍然可以进行其他功能。

## 1.3 类SemiSSTSCJPanel

实现半监督短文本数据流分类界面，同时检测输入参数是否正确，与有监督短文本数据流分类界面类似，以下简单介绍相关成员函数：

public SemiSSTSCJPanel (int width, int height)

——构造函数，输入参数为整个界面的宽和高。

private void initLeftJpanel()

——实现左边的结果输出界面。

private void initRightJpanel()

∟private Component loadDataJpanel()

private Component classifyJpanel()

——实现左边的参数设置界面，包括数据加载和分类的参数设置。

class startListener

——继承类ActionListener，用于响应“开始训练”按钮

∟private boolean checkFormat()

——检查输入参数格式是否正确，如果错误则向界面返回错误信息

class SemiSSTSCThread

——继承类Thread，是响应“开始训练”按钮的线程，用于与界面实现的线程分开，使得在响应开始训练事件时仍然可以进行其他功能。

# 第二章 有监督短文本数据流分类类库

## 2.1 概述

实现有监督短文本数据流分类的逻辑部分，该逻辑部分涉及到LDA主题模型的推断，所以本章除了介绍有监督短文本数据流分类的逻辑部分的代码实现外，还会介绍LDA主题模型的相关代码。表2.1给出了相关类的功能描述。

|  |  |
| --- | --- |
| 表2.1 有监督短文本数据流分类类的功能描述 | |
| 类名称 | 功能描述 |
| ShortTextStreamClassification类 | 实现有监督短文本数据流分类的逻辑部分 |
| ShortTextExpansion类 | 实现短文本数据流扩展 |
| DistanceCalculation类 | 实现短文本的语义距离计算 |
| EnsembleClassification类 | 实现集成分类 |
| ShortTexts类 | 存储短文本数据流信息 |
| LDA类 | 推断短文本数据流的主题概率 |
| Constants类 | JGibbLDA的常量类 |
| Conversion类 | JGibbLDA的中间文件名称设置类 |
| Dictionary类 | JGibbLDA的字典类 |
| Document类 | JGibbLDA的文本存储类 |
| Inferencer类 | JGibbLDA的主题推断类 |
| LDACmdOption类 | JGibbLDA的参数设置类 |
| LDADataset类 | JGibbLDA的数据集类 |
| Model类 | JGibbLDA的模型类 |
| Pair类 | JGibbLDA的工具类 |

## 2.2 类ShortTextStreamClassification

实现有监督短文本数据流分类的整体逻辑，以下简单介绍相关成员函数：

public ShortTextStreamClassification(File shortTextsFile)

——构造函数，输入参数为短文本数据流的文件，用于从文件中获取短文本数据流同时初始化其他相关变量。

public boolean extendShortText()

——扩展短文本数据流

public boolean vectorShortText()

∟private void adjustVectorizationFormat()

∟private String getWordIndex(String doc, Map<String, Integer> wordMap)

private String[] classIdToString(List<Integer> classId)

private String[] classIdToString(List<Integer> classId)

private String[] classMapToString(Map<String, Integer> classMap)

private String[] wordMapToString(Map<String, Integer> wordMap)

public native void oBTM(int K, int W, double alpha, double beta,

String input\_dir, int n\_day, String res\_dir, int n\_iter, double lam)

——向量化扩展后的短文本数据流，其中向量化短文本数据流用到了online BTM主题模型，该主题模型采用Java的本地接口调用C版本的online BTM算法封装的动态链接库。为了符合该动态链接库的调用，会首先对扩展后的短文本数据流划分数据块并进行格式调整，oBTM函数的参数输入依次是：主题个数、不同词数、先验参数alpha和beta值、扩展后的短文本数据流目录、短文本数据块个数、结果存储目录、迭代数以及衰退值。

public boolean classifyShortText()

∟private svm\_parameter initClassifyParam()

——初始化SVM分类器的参数信息，并分类向量化后的短文本数据流

## 2.3 类ShortTextExpansion

实现短文本数据流的扩展，该类是在类ShortTextStreamClassification中的extendShortText()函数中进行调用的。相关成员函数的介绍如下：

public ShortTextExpansion(File externalCorpusFile, File externalIntervalFile)

——构造函数，输入参数为外部语料库和区间文件。

public void expansion()

∟private void LDAinfer()

——推断短文本数据流的主题信息

private void initExpansion()

∟private boolean readTwords()

private boolean readInterval()

——为短文本数据流的扩展进行准备，包括读取外部语料库中的每个主题下的前n个词和扩展时所需要的区间信息。

public int[] binary\_InsertSort(int[] key, double[] value)

private int extendableTimes(int m, int k)

private void addTopicIds(int m, int k, int n)

private void addTexts(int m, int k, int n)

——扩展短文本数据流

## 2.4 类EnsembleClassification

构建集成分类器并分类短文本数据流，该类是在ShortTextStreamClassification中的classifyShortText()函数中进行调用的，相关成员函数的介绍如下：

public EnsembleClassification(String fileDir, List<Integer> classId, svm\_parameter param)

——构造函数，输入参数为向量化后的短文本数据流的存储目录、短文本数据流的标签信息和SVM的初始化参数。

public void classify()

∟private double[] predict(Vector<Double> vy, Vector<svm\_node[]> vx)

——构造集成模型并预测短文本的标签信息，同时根据数据块之间的概念漂移结果更新集成模型。

## 2.5 类DistanceCalculation

计算新数据块与构建基模型的数据块之间的语义距离，用于集成模型中各基模型的权值设置和集成模型的更新。该类是在EnsembleClassification中的classify()函数中进行调用的，相关成员函数的介绍如下：

public DistanceCalculation(svmModel model, Vector<svm\_node[]> test\_vx)

——构造函数，输入参数为基模型和新数据块

public void compute()

——计算新数据块与构建基模型的数据块之间的语义距离

∟private double cosine(svm\_node[] x, svm\_node[] y)

——计算两个短文本之间的余弦距离。

public double[] getPredictWeight()

——得到设置基模型的权值

## 2.6 类ShortTexts

存储短文本数据流的相关信息，包括原始短文本数据流，扩展后的短文本数据流、向量化的短文本数据流等信息。该类只有两个成员函数：

public void getOriginalShortTexts(List<String> data)

——得到原始短文本数据流

public void getExtendedShortTexts(String[] extendedTexts)

——得到扩展后的短文本数据流

## 2.7 类LDA

根据外部语料库，推断原始短文本数据流的主题概率。该类是LDA的一个使用类，该类的成员函数如下：

public LDA(File modelDir, List<String> data)

∟private void init()

——推断短文本数据流的主题概率

public double[][] getInfTheta()

——获得推断后的短文本数据流的文档-主题概率

public int getK()

——获得主题数

## 2.7 类Inferencer

LDA的主题推断类，相关成员函数介绍如下：

public boolean init(LDACmdOption option)

——初始化原有的主题模型

public Model inference( LDADataset newData)

——推断新数据的主题概率

public Model inference(String [] strs)

——推断新数据的主题概率

public Model inference(List<String> list)

——推断新数据的主题概率

public Model inference()

——推断新数据的主题概率

protected int infSampling(int m, int n)

——Gibbs采样

protected void computeNewTheta()

——计算被推断数据的文档-主题概率

protected void computeNewPhi()

——计算被推断数据的主题-词概率

protected void computeTrnTheta()

——计算原有数据的文档-主题概率

protected void computeTrnPhi()

——计算原有数据的主题-词概率

# 第三章 半监督短文本数据流分类类库

## 3.1 概述

实现半监督短文本数据流分类的逻辑部分，表3.1给出了相关类的功能描述。

|  |  |
| --- | --- |
| 表3.1半监督短文本数据流分类类的功能描述 | |
| 类名称 | 功能描述 |
| SemiShortTextStreamClassification类 | 实现半监督短文本数据流分类的整体逻辑部分 |
| Doc2Vector类 | 向量化原始短文本数据流 |
| SemiEnsembleClassifier类 | 半监督分类器 |
| SemiShortTexts类 | 存储部分有标签的短文本数据流信息 |
| InstanceConstruction类 | 用于构建聚类模型数据 |
| ClusterDistanceCalculation类 | 用于计算语义距离 |
| Classifier类 | 基分类器 |
| Cluster类 | 基聚类模型 |
| ClusterInfo类 | 存放簇信息 |

## 3.2 类SemiShortTextStreamClassification

实现半监督短文本数据流分类的整体逻辑，以下简单介绍相关成员函数：

public SemiShortTextStreamClassification(JtextArea printResultJTextArea, String shortTextFileName, String labelFileName)

——构造函数，输入参数为结果输出的文本组件、短文本数据流文件以及标签信息。初始化部分有标签短文本数据流以及其他信息。

public int vectorShortTexts(int currentProgress)

——向量化短文本数据流并更新程序的完成度，输入参数为上一个程序的完成进程。

public int classifyShortTexts(int currentProgress, int MAX\_PROGRESS)

——分类向量化的短文本数据流并更新程序的完成度，输入参数为上一个程序的完成进程和设置的最大进程。

## 3.3 类Doc2Vector

借助外部语料库的原始词向量集合将原始短文本数据流中的每个短文本表示为一组特定维度的向量。该类是在类SemiShortTextStreamClassification中的vectorShortTexts (int currentProgress)函数中进行调用的。相关成员函数的介绍如下：

public Doc2Vector(String externalCorpusFileName)

∟private void init()

——构造函数，输入参数为外部语料库文件，实现外部语料库信息（即原始词向量集合）的读取。

public void run(SemiShortTexts shortTexts, int chunkNum, int chunkSize)

——根据读取的原始词向量集合实现短文本数据流的特征表示。输入参数为原始短文本数据流，原始短文本数据流被划分的数据块个数和数据块大小。

## 3.4 类SemiEnsembleClassifier

构建半监督集成模型并预测短文本数据流，该类是在SemiShortTextStreamClassification中的classifyShortTexts(int currentProgress, int MAX\_PROGRESS)函数中进行调用的，相关成员函数的介绍如下：

public SemiEnsembleClassifier(int classifierNum, int clusterNum)

——构造函数，输入参数为分类器个数和聚类器个数。

public void init()

∟private void initParam()

——初始化聚类模型的数据格式和SVM的分类参数。

public double classify(List<String> data, List<Integer> classId, String label)

——构建半监督集成模型并计算新数据块与构建集成模型的数据块之间的语义相似性用于判断是否发生概念漂移，根据漂移检测结果预测新的数据块。输入参数为新短文本数据块，数据块标签和该数据块是否有标签的标识。

∟private double getAccuracy(double[][] predictResults, Vector<Double> vy)

——计算分类准确率，输入参数为新数据块的标签预测结果和新数据块的原始标签信息。

private ClusterInfo[] getClusterInfo(SimpleKMeans skmeans, Instances data, int[] instsCluster)

——计算簇信息，输入参数为聚类模型，构建聚类模型的数据，以及短文本属于哪个聚类簇的标识。

private double calculateEuclideanDist(double[] ds, Instance instance)

——计算两个簇之间的欧氏距离，输入参数为两个簇的不同表示方式。

## 3.5 类SemiShortTexts

存储半监督短文本数据流的相关信息。该类只有两个成员函数：

public void getOriginalShortTexts(List<String> data)

——得到原始短文本数据流

public void getLabelInfo(List<String> data)

——得到是否有标签的信息

## 3.6 类InstanceConstruction

将短文本数据块转化为可构造聚类模型的数据格式，相关成员函数如下：

public InstanceConstruction(int featureNum, List<String> classLabel)

∟private void init()

——构造函数，初始化数据格式，输入参数为特征数和类标签信息。

public Instances getInstances(List<String> data, List<Integer> classId)

——将短文本数据块转化为可构造聚类模型的数据格式，输入为数据块和对应的标签信息。

## 3.7 类ClusterDistanceCalculation

计算两个数据块的簇集合之间的语义距离，该类是在SemiEnsembleClassifier中的classify(List<String> data, List<Integer> classId, String label)函数中进行调用的，相关成员函数的介绍如下：

public ClusterDistanceCalculation(int newInstNum, ClusterInfo[] newClusterInfo, int instNum, ClusterInfo[] clusterInfos)

——构造函数，初始化相关信息。输入参数包括两个数据块的短文本数和簇集合。

public void compute()

∟private double calculateSimilarity(ClusterInfo ci1, ClusterInfo ci2)

∟private double calculateEuclideanDist(double[] ds, double[] es)

——计算两个数据块的簇集合之间的语义距离。

## 3.8 类Classifier

用于存储基分类器的类，该类继承了类svmModel，有关该类的介绍将会在第四章展开，相关成员函数如下：

public Classifier(svm\_parameter param, int max\_index, Vector<Double> vy, Vector<svm\_node[]> vx, int classNum, int featureNum)

——构造函数，用于训练分类器，根据类标签划分数据块获取簇集合。

∟private void getClusterInfos()

∟private void getClusterCentroids(double[][] clusterCentroids)

private void getClusterRadius(double[][] clusterCentroids, double[] clusterRadius)

∟private double calculateEuclideanDist(double[] ds, svm\_node[] nodes)

——根据类标签划分数据块获取簇集合。

public double[][] predict(Vector<Double> tvy, Vector<svm\_node[]> tvx)

——根据基分类器预测短文本的类标签。

## 3.9 类Cluster

用于存储基聚类模型的类，相关成员函数如下：

public Cluster(SimpleKMeans cluster, ClusterInfo[] clusterInfos, int instNum)

——构造函数，初始化基聚类器。

public double[][] predict(Instances data)

——根据基聚类器预测短文本的聚类簇ID

public void propagateLabel(LinkedList<Classifier> classifiers, int classNum)

——由基分类器模型向聚类模型中的每个聚类簇传递标签信息，使得每个聚类簇也可以直接预测短文本的类标签信息。

# 第四章 通用类库

## 4.1 概述

整个软件的通用类库，表4.1给出了相关类的功能描述。

|  |  |
| --- | --- |
| 表4.1 通用类库 | |
| 类名称 | 功能描述 |
| FileTools类 | 文件处理类 |
| OperationTools类 | 运算类 |
| svm\_predict类 | SVM的预测类 |
| svm\_train类 | SVM的训练类 |
| svmModel类 | 存储SVM模型类 |
| SVMTools类 | SVM相关操作类 |
| TypeTools类 | 类型判断类 |

## 4.2 类FileTools

用于文件的读写，相关成员函数如下：

public static List<String> readFile(String fileName)

——读文件，输入为文件名，返回数据链表

∟public static List<String> readFile(File file)

——读文件，输入为File类型文件，返回数据链表

public static void writeFiles(String dir, String[][] data)

——写多个文件，输入为文件名和被写的数据

∟public static void writeFile(String filename, String[] data)

——写文件，输入为文件名和被写入数据

public static void isChartPathExist(String dirPath)

——判断文件路径是否存在，输入为文件夹

## 4.3 类OperationTools

用于数据的运算，相关成员函数如下：

public static void add2DimArr(double[][] A, double[][] B)

——将二维数组A和二维数组B相加，并将结果存于数组A。

public static double[] addColumn(double[][] A)

——将二维数组A按列相加，返回相加结果。

public static void zeroToOne2(double[][] A)

——将二维数组A中所有的零元素替换成1，结果存于数组A

public static void zeroToOne1(double[] A)

——将一维数组A中所有的零元素替换成1，结果存于数组A

public static void division(double[][] A, double[] B)

——将二维数组A中每行的任一元素除以一维数组B中对应行的元素

public static double maxValue(double[] arr)

——返回一维数组arr中最大的元素值

public static double minValue(double[] arr)

——返回一维数组arr中最小的元素值

public static double maxInx(double[] arr)

——返回一维数组arr中最大的元素值的索引

public static int minInx(double[] arr)

——返回一维数组arr中最小的元素值的索引

## 4.4 类svm\_predict

SVM的预测，相关成员函数介绍如下：

public svm\_predict(svm\_model model, int classNum)

——构造函数，初始化SVM模型

public void predict(Vector<Double> tvy, Vector<svm\_node[]> tvx)

——预测新数据

## 4.5 类svm\_train

SVM的训练，相关成员函数介绍如下：

public svm\_train(svm\_parameter param)

——构造函数，初始化模型训练参数

public void run(Vector<svm\_node[]> vx, Vector<Double> vy, int max\_index)

——训练SVM分类模型

## 4.6 类svmModel

存储SVM模型类，相关成员函数介绍如下：

public svmModel(svm\_model model, int timestamps, Vector<Double> vy, Vector<svm\_node[]> vx)

∟public void divideLabel()

——构造函数，初始化SVM模型，并根据类标签划分构建模型的数据。

## 4.7 类SVMTools

SVM相关操作类，相关成员函数介绍如下：

public static double atof(String s)

——将字符串转化为double型

public static int atoi(String s)

——将字符串转化为int型

public static int getClassifyData(List<String> data, List<Integer> ids, Vector<Double> vy, Vector<svm\_node[]> vx)

——将需分类数据data和对应的标签信息转化为符合SVM分类的数据，并存于后两个参数中。

## 4.8 类TypeTools

类型判断类，相关成员函数介绍如下：

public static boolean isInteger(String str, int s, int e)

——判断字符串是否是大于等于s小于等于e的整数

public static boolean isdigit(String str)

——判断字符串是否是数字

public static boolean isDouble(String str)

——判断字符串是否是double型数