INSPER

Ciência dos Dados

Projeto 3

Classificação e Clustering de sons urbanos

Vitor Eller, Henry Rocha, André Weber

São Paulo

2018

**INTRODUÇÃO**

O objetivo do projeto é fazer um clustering de um banco de dados de áudios relacionados a sons urbanos com 40 classes ao todo. Dessa forma, utilizamos as bibliotecas, pandas, numpy, matplotlib, librosa, scipy e scikit-learn

O dataset foi criado através da análise de cada áudio com 52 ferramentas de 6 features ao todo e em seguida calculada a acurácia de modelos como KNeighbors, RandomForestClassifier e GradientBoostingClassifier.

**DESENVOLVIMENTO**

* **Bibliotecas**

A biblioteca **librosa**: tem como enfoque a análise de músicas e áudios. Foi criada para ter uma alta relação com numpy, scipy e MATLAB, facilitando no uso.

A utilização dela no projeto foi crucial, pois assim carregamos os áudios a partir da função librosa.load(). Assim como a extração das features usadas, librosa.feature.feature\_name().

A biblioteca **pandas**:Foi utilizada basicamente para manusear os dataframes de treino e teste.

A biblioteca **numpy**: Também foi de extrema importância, pois através de suas funções conseguimos fazer principalmente a transformação de Fourier np.fft.fft(audio). Os valores como média, mediana, percentil, entre outros, também foram obtidos a partir da biblioteca.

A biblioteca **matplotlib**: Utilizada a extensão pyplot para elaboração de gráficos.

A biblioteca **scipy**: Também utilizada para fazer a Transformação de Fourier, scipy.fft(waveform). E para obter o valor de amplitude interquartil scipy.stats.iqr().

A biblioteca **scikit-learn**: Basicamente para processar o dataset de treino e teste para em seguida utilizar a aplicação de machine learning com os modelos e obter a acurácia do programa. RandomForestClassifier() | acc(x,y).

* **Processo**

Resumidamente, o processo se iniciou com o load de todos áudios, obtendo a onda e o SampleRate (dado 44100 previamente), que é a quantidade de amostras do sinal coletadas em função do tempo. Em seguida, foi feita a Transformação de Fourier para decompor a onda em partes de senoides, assim em vez de guardar uma alta quantidade de dados, obtém-se a frequência que será usada para recriar a onda original.

Após a Transformada conseguimos obter as features e seus valores. Extraímos informações em cima de Fourier, harmônico, percussivo, spectral centroid, spectral flatness, spectral contrast. A partir delas foram obtido os valores: mediana, média, primeiro percentil, segundo percentil, amplitude interquartil, mínimo, máximo, desvio padrão e batidas por minuto.

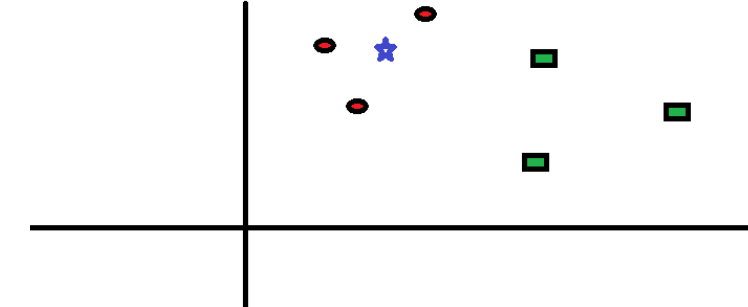
Depois de organizar o dataset e normalizar os dados, foram calculadas as acurácias com três modelos para comparação, KNeighbors, RandomForest, GradientBoosting.

* **Features**

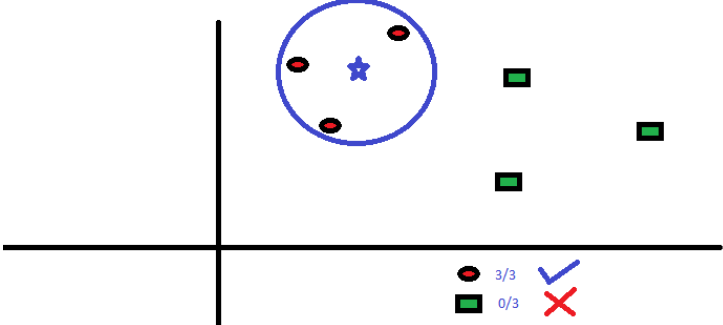
1. **Fourier –** Informações sobre a frequência da onda original.
2. **Harmônico** – Frequência de vibração com propriedade de causar ressonância. Correspondem às notas e escala musical.
3. **Percussivo** – Sons e sinais obtidos através de impacto raspagem ou agitação.
4. **Spectral centroid (centróide espectral)** – Indica onde o centro de massa do espectro está localizado. (Calculado com a média ponderada das frequências)**.**
5. **Spectral flatness (nivelamento espectral)** – Medido em decibéis. Indica o quanto o som se aproxima de ser barulho puro. Quantifica “noise-like”
6. **Spectral contrast (contraste espectral)** – Como o próprio nome já diz, representa uma distribuição relativa espectral e suas características em um áudio.

* **Classificadores**

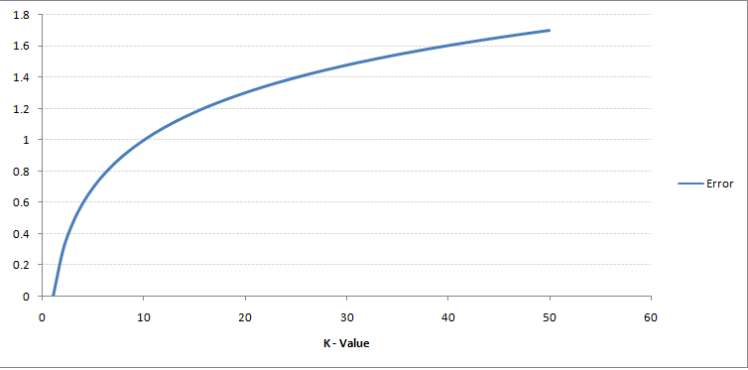
**KNeighbors** – Este classificador se baseia no conceito de vizinhança. Ou seja, gostaríamos de classificar um elemento, assim definimos uma constante que selecionará a quantidade de vizinhos que serão escolhidos nas redondezas deste elemento. A classe com mais membros na vizinhança determinará o objeto em questão.

 Para estabelecer uma constante ideal, é necessário ver a curva de taxa de erro em função do valor de K e a curva de validação de erro em função de K (após separar dataset treino e validação).

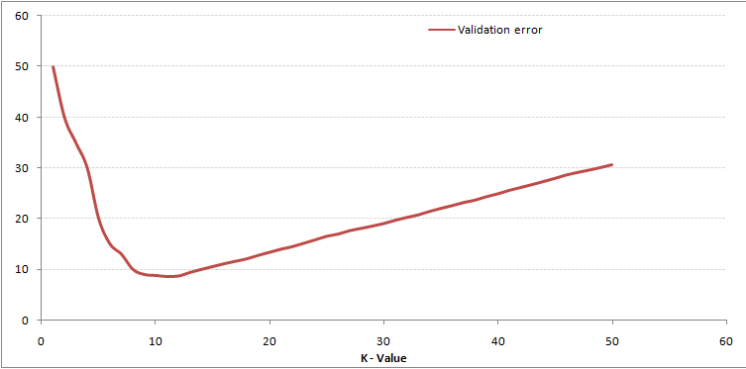
*Estrela azul: elemento a ser classificado; Círculo vermelho: Classe 1; Retângulo verde: Classe 2;*



*Definida a constante como 3, assim definindo os três mais próximos elementos.*

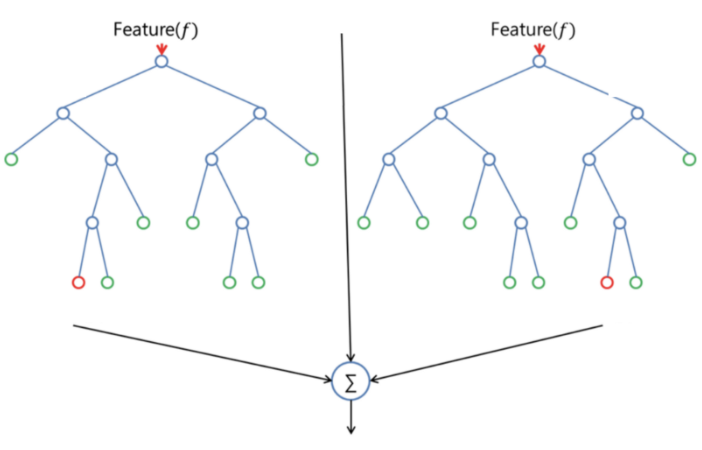


*Curva da taxa de erro em função do valor-K.*

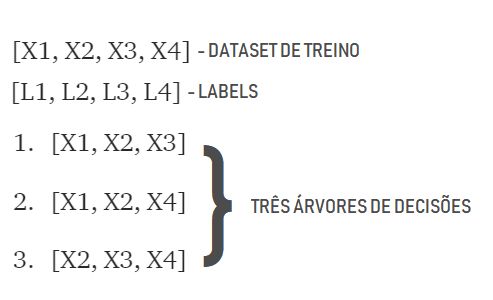


*Curva de validação de erro em função do valor-K.*

**RandomForest** – Este classificador basicamente se baseia em um modelo de decisão por “árvore”. Assim, estabelece diversas combinações com resultados como predições que são fundidos e estabelece uma predição final, que pode ser considerada como a média de todas. O algoritmo aplica uma aleatoriedade nas combinações e nas features, que posteriormente é selecionada a melhor.



*Random Forest com duas árvores.*



*Random Forest criou 3 árvores de decisões e então preveria o final a partir do resultado de cada árvore, tendo o mais comum com mais relevância.*

**GradientBoosting** – A lógica por trás deste classificador está em parte na regressão linear, em que a soma dos resíduos é zero. Porém, se houver diferentes de zeros, é possível que haja padrões nesses resíduos que podem ser promovidos para melhorar o modelo. Assim, o processo é feito até que todos resíduos fiquem aleatoriamente distribuídos. Chegando neste ponto, não precisa mais modelar eles.