Nama: Hifni khakim

Kelas: TI KIP C2 Prioritas

NIM: 41226198

Prodi: Teknik Informatika

Tugas Data mining pertemuan 2

Preprocessing Data (Normalization, Missing value, Imbalance data,Feature selection)

Fase prapemrosesan data adalah bagian ilmu data yang paling menantang dan memakan waktu, tetapi juga merupakan salah satu bagian terpenting. Jika Anda gagal membersihkan dan menyiapkan data, hal itu dapat membahayakan model.

Saat berhadapan dengan data dunia nyata, Ilmuwan Data akan selalu perlu menerapkan beberapa teknik prapemrosesan agar data lebih bermanfaat. Teknik-teknik ini akan memfasilitasi penggunaannya dalam machine learning (ML) algorithms, mengurangi kompleksitas untuk mencegah overfitting, dan menghasilkan model yang lebih baik.

Karena itu, mari masuk ke ikhtisar tentang apa itu preprocessing data, mengapa itu penting, dan pelajari teknik utama yang digunakan dalam fase kritis ilmu data ini. Inilah semua yang akan kami bahas dalam panduan ini:

**Apa itu Pemrosesan Data?**

Setelah memahami nuansa kumpulan data Anda dan masalah utama dalam data melalui Analisis Data Eksplorasi, prapemrosesan data mulai berlaku dengan menyiapkan kumpulan data Anda untuk digunakan dalam model.

Di dunia yang ideal, kumpulan data Anda akan sempurna dan tanpa masalah. Sayangnya, data dunia nyata akan selalu menghadirkan beberapa masalah yang perlu Anda tangani. Pertimbangkan, misalnya, data yang Anda miliki di perusahaan Anda. Dapatkah Anda memikirkan ketidakkonsistenan seperti kesalahan ketik, data yang hilang, skala yang berbeda, dll.? Contoh-contoh ini sering terjadi di dunia nyata dan perlu disesuaikan agar data lebih bermanfaat dan mudah dipahami.

Proses ini, di mana kami membersihkan dan menyelesaikan sebagian besar masalah dalam data, adalah apa yang kami sebut langkah prapemrosesan data.

**Mengapa Data Preprocessing Penting?**

Jika Anda melewatkan langkah pra-pemrosesan data, ini akan memengaruhi pekerjaan Anda nanti saat menerapkan kumpulan data ini ke model pembelajaran mesin. Sebagian besar model tidak dapat menangani nilai yang hilang. Beberapa di antaranya dipengaruhi oleh outlier, dimensi tinggi, dan data berisik, jadi dengan melakukan prapemrosesan data, Anda akan membuat kumpulan data lebih lengkap dan akurat. Fase ini sangat penting untuk melakukan penyesuaian yang diperlukan pada data sebelum memasukkan dataset ke dalam model pembelajaran mesin Anda.

**Teknik Preprocessing Data Penting**

Sekarang setelah Anda mengetahui lebih banyak tentang fase prapemrosesan data dan mengapa itu penting, mari kita lihat teknik utama untuk diterapkan dalam data, membuatnya lebih bermanfaat untuk pekerjaan kita di masa mendatang. Teknik yang akan kita jelajahi adalah:

* Data Cleaning
* Dimensionality Reduction
* Feature Engineering
* Sampling Data
* Data Transformation
* Imbalanced Data

### Data Cleaning

Salah satu aspek terpenting dari fase prapemrosesan data adalah mendeteksi dan memperbaiki pengamatan yang buruk dan tidak akurat dari kumpulan data Anda untuk meningkatkan kualitasnya. Teknik ini mengacu pada identifikasi nilai yang tidak lengkap, tidak akurat, terduplikasi, tidak relevan atau nol dalam data. Setelah mengidentifikasi masalah ini, Anda perlu mengubah atau menghapusnya. Strategi yang Anda adopsi bergantung pada domain masalah dan tujuan proyek Anda. Mari kita lihat beberapa masalah umum yang kita hadapi saat menganalisis data dan cara menanganinya.

#### **Noisy Data**

Biasanya, data yang berisik mengacu pada data yang tidak berarti dalam kumpulan data Anda, catatan yang salah, atau pengamatan yang digandakan. Misalnya, bayangkan ada kolom di database Anda untuk 'usia' yang memiliki nilai negatif. Dalam hal ini, pengamatan tidak masuk akal, jadi Anda dapat menghapusnya atau menyetel nilainya sebagai nol (kami akan membahas cara memperlakukan nilai ini di bagian "Data Hilang").

Kasus lainnya adalah ketika Anda perlu menghapus data yang tidak diinginkan atau tidak relevan. Misalnya, Anda perlu memprediksi apakah seorang wanita hamil atau tidak. Anda tidak memerlukan informasi tentang warna rambut, status perkawinan, atau tinggi badan mereka, karena tidak relevan dengan modelnya.

Sebuah outlier dapat dianggap sebagai noise, meskipun itu mungkin merupakan catatan yang valid, tergantung pada outliernya. Anda harus menentukan apakah outlier dapat dianggap sebagai data derau dan apakah Anda dapat menghapusnya dari kumpulan data atau tidak.

solusi:

Teknik umum untuk data noise adalah pendekatan binning, di mana Anda mengurutkan nilai terlebih dahulu, kemudian membaginya menjadi "tempat sampah" (ember dengan ukuran yang sama), dan kemudian menerapkan rata-rata/median di setiap tempat sampah, menghaluskannya. Jika Anda ingin mempelajari lebih lanjut, berikut adalah artikel bagus tentang menangani data kebisingan.

#### **Missing Data**

Masalah umum lainnya yang kami hadapi dalam data dunia nyata adalah tidak adanya titik data. Sebagian besar model pembelajaran mesin tidak dapat menangani nilai yang hilang dalam data, jadi Anda perlu mengintervensi dan menyesuaikan data agar dapat digunakan dengan benar di dalam model. Ada beberapa pendekatan berbeda yang dapat Anda ambil untuk menanganinya (biasanya disebut imputasi):

Solusi 1 :

Solusi paling sederhana adalah menghapus pengamatan itu. Namun, ini hanya disarankan jika:

1) Anda memiliki kumpulan data yang besar dan beberapa catatan yang hilang, jadi menghapusnya tidak akan memengaruhi distribusi kumpulan data Anda.

2) Sebagian besar atribut dari observasi tersebut adalah null, sehingga observasi itu sendiri tidak memiliki arti.

Solusi 2 :

Solusi lainnya adalah dengan menggunakan konstanta global untuk mengisi celah tersebut, seperti “NA” atau 0, tetapi hanya jika sulit untuk memprediksi nilai yang hilang. Pilihan alternatifnya adalah menggunakan rata-rata atau median dari atribut tersebut untuk mengisi kesenjangan.

Solusi 3 :

Menggunakan metode pengisian backward/forward adalah pendekatan lain yang dapat diterapkan, di mana Anda mengambil nilai sebelumnya atau berikutnya untuk mengisi nilai yang hilang.

Solusi 4 :

Pendekatan yang lebih kuat adalah penggunaan algoritme pembelajaran mesin untuk mengisi poin data yang hilang ini. Sebagai contoh:

Dengan menggunakan KNN, pertama-tama temukan k instance yang lebih dekat dengan instance nilai yang hilang, lalu dapatkan rata-rata atribut yang terkait dengan k-nearest neighbor (KNN).

Dengan menggunakan regresi, untuk setiap atribut yang hilang, pelajari regresi yang dapat memprediksi nilai yang hilang ini berdasarkan atribut lainnya.

Tidak mudah untuk memilih teknik khusus untuk mengisi missing value dalam kumpulan data kami, dan pendekatan yang Anda gunakan sangat bergantung pada masalah yang sedang Anda kerjakan dan jenis missing value yang Anda miliki.

Topik ini melampaui cakupan artikel ini, namun perlu diingat bahwa kita dapat memiliki tiga jenis nilai yang hilang, dan masing-masing harus diperlakukan secara berbeda:

Tipe 1: Hilang Sepenuhnya Secara Acak (Missing Completely at Random =MCAR)

Tipe 2: Hilang Secara Acak (Missing at Random = MAR)

Tipe 3: Hilang Tidak Secara Acak (Missing Not at Random = MNAR)

Jika Anda terbiasa dengan Python, pustaka sklearn memiliki alat yang berguna untuk langkah prapemrosesan data ini, termasuk KNN Imputer yang saya sebutkan di atas.

#### **Structural Errors**

Kesalahan struktural biasanya merujuk pada beberapa kesalahan ketik dan ketidakkonsistenan nilai data.

Misalnya, ada pasar dan kami menjual sepatu di situs web kami. Data tentang produk yang sama dapat ditulis dengan cara berbeda oleh penjual berbeda yang menjual sepatu yang sama. Bayangkan salah satu atribut yang kita miliki adalah merek sepatu tersebut, dan gabungkan nama merek untuk sepatu yang sama yang kita miliki: Nike, nike, NIKE. Kami perlu memperbaiki masalah ini sebelum memberikan data ini ke model, jika tidak, model dapat memperlakukannya sebagai hal yang berbeda. Dalam hal ini, perbaikannya mudah: cukup ubah semua kata menjadi huruf kecil. Namun, mungkin diperlukan perubahan yang lebih kompleks untuk memperbaiki ketidakkonsistenan dan kesalahan ketik dalam skenario lain.

Masalah ini umumnya memerlukan intervensi manual daripada menerapkan beberapa teknik otomatis.

### Dimensionality Reduction

Pengurangan dimensi berkaitan dengan pengurangan jumlah fitur input dalam data pelatihan.

**The Curse of Dimensionality in Your Dataset**

Dengan set data dunia nyata, biasanya ada banyak sekali atribut, dan jika kami tidak mengurangi jumlah ini, ini dapat memengaruhi performa model nantinya saat kami memberinya set data ini. Mengurangi jumlah fitur sambil mempertahankan sebanyak mungkin variasi dalam dataset akan berdampak positif dalam banyak hal, seperti:

Membutuhkan lebih sedikit sumber daya komputasi

Meningkatkan kinerja model secara keseluruhan

Mencegah overfitting (ketika model menjadi terlalu kompleks dan model menghafal data pelatihan, alih-alih belajar, sehingga dalam data pengujian kinerjanya menurun drastis)

Menghindari multikolinearitas (korelasi yang tinggi dari satu atau lebih variabel bebas). Juga, menerapkan teknik ini akan mengurangi data kebisingan.

Mari selami jenis utama pengurangan dimensi yang dapat kita terapkan pada data kita untuk membuatnya lebih baik untuk digunakan nanti.

#### **Feature Selection**

Pemilihan fitur mengacu pada proses pemilihan variabel (fitur) terpenting yang terkait dengan variabel prediksi Anda, dengan kata lain, memilih atribut yang paling berkontribusi pada model Anda. Berikut adalah beberapa teknik untuk pendekatan ini yang dapat Anda terapkan baik secara otomatis maupun manual:

Korelasi Antara Fitur: Ini adalah pendekatan yang paling umum, yang menghilangkan beberapa fitur yang memiliki korelasi tinggi dengan yang lain.

Uji Statistik: Alternatif lain adalah menggunakan uji statistik untuk memilih fitur, memeriksa hubungan setiap fitur secara individual dengan variabel keluaran. Ada banyak contoh di perpustakaan scikit-learn seperti SelectKBest, SelectPercentile, chi2, f\_classif, f\_regression.

Recursive Feature Elimination (RFE): Penghapusan Fitur Rekursif, juga dikenal sebagai Penghapusan Mundur, di mana algoritme melatih model dengan semua fitur dalam kumpulan data, menghitung kinerja model, lalu menghapus satu fitur pada satu waktu, berhenti saat peningkatan kinerja menjadi diabaikan.

Ambang Batas Varians: Metode pemilihan fitur lainnya adalah ambang varians, yang mendeteksi fitur dengan variabilitas tinggi di dalam kolom, memilih fitur yang melebihi ambang batas. Premis dari pendekatan ini adalah bahwa fitur dengan variabilitas rendah di dalamnya memiliki pengaruh yang kecil terhadap variabel keluaran.

Selain itu, beberapa model secara otomatis menerapkan pemilihan fitur selama pelatihan. Model berbasis pohon keputusan dapat memberikan informasi tentang pentingnya fitur, memberi Anda skor untuk setiap fitur data Anda. Semakin tinggi nilainya, semakin relevan untuk model Anda. Untuk lebih banyak algoritme yang diterapkan di sklearn, pertimbangkan untuk memeriksa modul feature\_selection.

#### **Linear Methods**

Seperti namanya, metode linier menggunakan transformasi linier untuk mengurangi dimensi data.

Pendekatan yang paling umum: Analisis Komponen Utama (PCA, dalam hal efisiensi memori dan data jarang, Anda dapat menggunakan IncrementalPCA atau SparsePCA), sebuah metode yang mengubah fitur asli di ruang dimensi lain menangkap banyak variabilitas data asli dengan jauh lebih sedikit variabel. Namun, fitur baru yang diubah kehilangan interpretasi dari data asli, dan hanya bekerja dengan variabel kuantitatif.

Jenis metode linier lainnya adalah Analisis Faktor dan Analisis Diskriminan Linier.

**Non-Linear Methods**

The non-linear methods (or manifold learning methods) digunakan ketika data tidak cocok dalam ruang linier. Gagasan di balik teknik ini adalah bahwa dalam ruang berdimensi tinggi, sebagian besar fitur penting terletak pada sejumlah kecil manifold berdimensi rendah. Banyak algoritma menggunakan pendekatan ini.

**Multi-Dimensional Scaling**

Multi-Dimensional Scaling (MDS) adalah salah satunya, dan menghitung jarak antara setiap pasangan objek dalam ruang geometris. Algoritme ini mengubah data ke dimensi yang lebih rendah, dan pasangan yang dekat di dimensi yang lebih tinggi tetap berada di dimensi yang lebih rendah juga.

The Isometric Feature Mapping (Isomap) adalah perpanjangan dari MDS, tetapi alih-alih jarak Euclidean, ia menggunakan jarak geodesik.

Contoh lain dari metode non-linear adalah Locally Linear Embedding (LLE), Spectral Embedding, t-distributed Stochastic Neighbor Embedding (t-SNE)

**Feature Engineering: Using Domain Knowledge to Create Features**

Pendekatan rekayasa fitur digunakan untuk membuat fitur yang lebih baik untuk kumpulan data Anda yang akan meningkatkan kinerja model. Kami terutama menggunakan pengetahuan domain untuk membuat fitur tersebut, yang kami hasilkan secara manual dari fitur yang ada dengan menerapkan beberapa transformasi pada fitur tersebut. Berikut adalah beberapa contoh dasar yang dapat Anda terapkan dengan mudah ke kumpulan data untuk berpotensi meningkatkan kinerja model Anda:

**Decompose Categorical Attributes**

Contoh pertama adalah penguraian atribut kategorikal dari kumpulan data Anda. Bayangkan Anda memiliki fitur dalam data Anda tentang warna rambut dan nilainya cokelat, pirang, dan tidak diketahui. Dalam hal ini, Anda dapat membuat kolom baru yang disebut "berwarna" dan menetapkan 1 jika Anda mendapatkan warna dan 0 jika nilainya tidak diketahui.

**Decompose a DateTime**

Contoh lain adalah mendekomposisi fitur datetime, yang berisi informasi berguna, tetapi sulit bagi model untuk memanfaatkan bentuk asli data. Jadi, jika menurut Anda masalah Anda memiliki ketergantungan waktu, dan Anda mungkin menemukan beberapa hubungan antara datetime dan variabel keluaran, luangkan waktu untuk mencoba mengonversi kolom datetime tersebut menjadi fitur yang lebih mudah dipahami untuk model Anda, seperti "periode hari, ” “hari dalam seminggu,” dan seterusnya.

**Reframe Numerical Quantities**

Contoh terakhir ini lebih tentang penanganan data numerik. Katakanlah Anda memiliki kumpulan data tentang beberapa pembelian pakaian untuk toko tertentu. Selain jumlah pembelian absolut, Anda mungkin tertarik untuk membuat fitur baru terkait musim pembelian tersebut. Jadi, Anda mungkin akan menambahkan empat kolom lagi ke kumpulan data Anda tentang pembelian di musim panas, musim dingin, musim gugur, dan musim semi. Bergantung pada masalah yang Anda coba selesaikan, ini dapat membantu Anda dan meningkatkan kualitas kumpulan data Anda.

Oleh karena itu, bagian ini lebih kepada menggunakan pengetahuan domain Anda tentang masalah tersebut untuk membuat fitur yang memiliki daya prediksi tinggi. Jika Anda ingin mempelajari lebih lanjut tentang ini, inilah blog yang bagus tentang rekayasa fitur.

### Handling a Large Amount of Data (Sampling Data)

Meskipun semakin banyak data yang Anda miliki, akurasi model cenderung semakin besar, beberapa algoritme pembelajaran mesin dapat mengalami kesulitan menangani data dalam jumlah besar dan mengalami masalah seperti saturasi memori, peningkatan komputasi untuk menyesuaikan parameter model, dan sebagainya. Untuk mengatasi masalah ini, berikut adalah beberapa teknik pengambilan sampel data yang dapat kita gunakan:

**Sampling without replacement**. Pendekatan ini menghindari pengulangan data yang sama dalam sampel, jadi jika record dipilih, record tersebut akan dihapus dari populasi.

**Sampling with replacement.** Dengan pendekatan ini, objek tidak dihilangkan dari populasi dan dapat diulang berkali-kali untuk data sampel karena dapat diambil lebih dari satu kali.

**Stratified sampling.** Metode ini lebih kompleks dan mengacu pada pemisahan data menjadi banyak partisi dan mendapatkan sampel acak untuk setiap partisi. Dalam kasus di mana kelas tidak proporsional, pendekatan ini mempertahankan jumlah kelas yang proporsional sesuai dengan data aslinya.

**Progressive sampling.** Teknik terakhir ini dimulai dengan ukuran kecil dan terus meningkatkan dataset sampai diperoleh ukuran sampel yang cukup.

**Data Transformation: Converting the Data to the Same Structure**

Salah satu langkah terpenting dalam fase preprocessing adalah transformasi data, yang mengubah data dari satu format ke format lainnya. Beberapa algoritme berharap data masukan diubah, jadi jika Anda tidak menyelesaikan proses ini, Anda mungkin mendapatkan kinerja model yang buruk atau bahkan membuat bias.

Sebagai contoh, model KNN menggunakan ukuran jarak untuk menghitung tetangga yang lebih dekat dengan record tertentu. Jika Anda memiliki fitur yang skalanya sangat tinggi dibandingkan dengan fitur lain dalam model Anda, maka model Anda akan cenderung menggunakan lebih banyak fitur ini daripada yang lain, sehingga menimbulkan bias pada model Anda. Beberapa teknik utama yang digunakan untuk menangani masalah ini adalah:

Transformation for categorical variables

Minmax scaler/ normalization

Standard scaler

Others: The maxAbs scaler, robust scaler, power transformer scaler

**Transformation for categorical variables**

Variabel kategorikal, biasanya diekspresikan melalui teks, tidak digunakan secara langsung di sebagian besar model pembelajaran mesin, jadi penting untuk mendapatkan pengkodean numerik untuk fitur kategorikal. Pendekatan yang Anda gunakan akan tergantung pada jenis variabel.

**Ordinal Variables**

Misalkan Anda memiliki data kualitatif ordinal, yang berarti ada keteraturan dalam nilai (seperti kecil, sedang, besar). Dalam hal ini, Anda perlu menerapkan fungsi pemetaan untuk mengganti string menjadi angka seperti: {“small”: 1, “medium”: 2, “large”: 3}. Anda dapat menggunakan kelas Label Encoder di sklearn, yang melakukannya untuk Anda.

**Nominal Variables**

Jika Anda memiliki variabel nominal dalam database Anda, yang berarti tidak ada urutan di antara nilai-nilai tersebut, Anda tidak dapat menerapkan strategi yang Anda gunakan dengan data ordinal. Teknik yang paling umum digunakan dengan jenis variabel ini adalah One Hot Encoding, yang mengubah satu kolom menjadi n kolom (di mana n mewakili nilai unik dari kolom asli), menetapkan 1 ke label di kolom asli dan 0 untuk yang lainnya. . Anda juga dapat menemukan orang yang menggunakan get\_dummies dari panda.

Misalnya, bayangkan kolom musim dengan empat label: Musim Dingin, Musim Semi, Musim Panas, dan Musim Gugur. Menerapkan enkode one-hot mengubahnya menjadi season\_winter, season\_spring, season\_summer, dan season\_autumn. Jika Anda memiliki nilai Musim Panas yang ditetapkan ke musim dalam catatan Anda, nilai tersebut akan diterjemahkan menjadi musim\_musim panas 1, dan tiga kolom lainnya akan menjadi 0.

Saat bekerja dengan One Hot Encoding, Anda perlu mengetahui masalah multikolinearitas. Solusi sederhana adalah menghapus salah satu kolom. Mengacu pada contoh di atas, jika kita memiliki season\_summer, season\_spring, dan season\_autumn sebagai 0, kita tahu ini musim dingin.

Anda dapat melihat di atas teknik utama untuk menangani transformasi data dengan data kualitatif, jadi sekarang mari kita lihat beberapa metode berbeda untuk data kontinu.

**Min-Max Scaler / Normalization**

Scaler min-max, juga dikenal sebagai normalisasi, adalah salah satu scaler yang paling umum dan mengacu pada penskalaan data antara rentang yang telah ditentukan (biasanya antara 0 dan 1). Masalah utama dengan teknik ini adalah sensitif terhadap outlier, tetapi layak digunakan saat data tidak mengikuti distribusi normal. Metode ini bermanfaat untuk algoritme seperti KNN dan Neural Networks karena tidak mengasumsikan distribusi data apa pun.

**Standard Scaler**

Scaler standar adalah teknik lain yang banyak digunakan yang dikenal sebagai normalisasi atau standardisasi z-score. Ini mengubah data sehingga rata-rata data adalah nol dan standar deviasinya adalah satu. Pendekatan ini bekerja lebih baik dengan data yang mengikuti distribusi normal dan tidak sensitif terhadap outlier.

**Other Scalers**

Min-max dan scaler standar adalah metode yang paling umum, tetapi banyak teknik berbeda yang mungkin berguna untuk aplikasi Anda, seperti:

Penskalaan maxAbs: Teknik ini mengambil nilai maksimum absolut dari fitur dan membagi setiap catatan dengan nilai maksimum ini, menskalakan data dalam kisaran -1 dan 1.

Penskala yang kuat: Teknik ini menghilangkan median dari data dan menskalakannya menggunakan interquartile range (IQR). Seperti namanya, itu kuat untuk outlier.

Scaler transformator daya: Teknik ini mengubah distribusi data, menjadikannya lebih seperti distribusi normal. Ini paling banyak digunakan dengan data heteroskedastisitas, yang berarti bahwa semua variabel tidak memiliki varian yang sama.

Bergantung pada masalah yang dihadapi, scaler yang berbeda akan membantu Anda meningkatkan hasil. Saya telah membuat daftar opsi yang paling umum, tetapi ada lebih banyak lagi yang dapat Anda temukan di sana.

**Handling Data with Unequal Distribution of Classes (Imbalanced Data)**

klasifikasi data adalah bahwa kelas tidak seimbang (salah satu kelas memiliki lebih banyak contoh daripada yang lain), menciptakan bias yang kuat untuk model tersebut.

Bayangkan Anda ingin memprediksi apakah suatu transaksi curang. Berdasarkan data pelatihan Anda, 95% kumpulan data Anda berisi catatan tentang transaksi normal, dan hanya 5% data Anda tentang transaksi penipuan. Berdasarkan itu, model Anda kemungkinan besar akan cenderung memprediksi kelas mayoritas, mengklasifikasikan transaksi penipuan sebagai transaksi normal.

Ada tiga teknik utama yang dapat kita gunakan untuk mengatasi kekurangan dalam kumpulan data ini:

1. **Oversampling**
2. **Undersampling**
3. **Hybrid**

**Oversampling**

Pendekatan oversampling adalah proses meningkatkan dataset Anda dengan data sintetis dari kelas minoritas. Teknik paling populer yang digunakan untuk ini adalah Teknik Oversampling Minoritas Sintetis (SMOTE). Singkatnya, dibutuhkan contoh acak dari kelas minoritas. Kemudian titik data acak lainnya dipilih melalui k-tetangga terdekat dari pengamatan pertama, dan rekor baru dibuat di antara dua titik data terpilih ini.

**Undersampling**

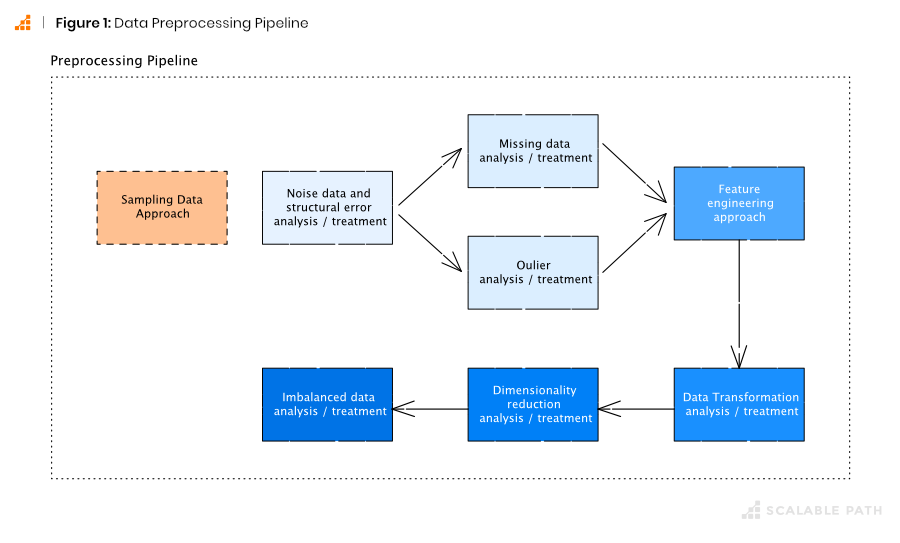
Teknik undersampling, sebaliknya, adalah proses mengurangi kumpulan data Anda dan menghapus data nyata dari kelas mayoritas Anda. Algoritme utama yang digunakan dalam pendekatan ini adalah TomekLinks, yang menghilangkan observasi berdasarkan tetangga terdekat, dan Edited Nearest Neighbors (ENN), yang menggunakan k-nearest neighbor bukan hanya satu seperti di Tomek.

**Hybrid**

Pendekatan hybrid menggabungkan teknik oversampling dan undersampling dalam kumpulan data Anda. Salah satu algoritma yang digunakan dalam metode ini adalah SMOTEENN, yang memanfaatkan algoritma SMOTE untuk oversampling pada kelas minoritas dan ENN untuk undersampling pada kelas mayoritas.

**The Data Preprocessing Pipeline**

secara umum, alur operasi prapemrosesan data berikut:



Langkah 1: Mulailah dengan menganalisis dan menangani kebenaran atribut, seperti mengidentifikasi data kebisingan dan kesalahan struktural apa pun dalam kumpulan data.

Langkah 2: Analisis data yang hilang, beserta outliernya, karena pengisian nilai yang hilang bergantung pada analisis outliernya. Setelah menyelesaikan langkah ini, kembali ke langkah pertama jika perlu, periksa ulang redundansi dan masalah lainnya.

Langkah 3: Proses penambahan pengetahuan domain ke dalam fitur baru untuk dataset Anda. Jika Anda tidak mendapatkan fitur baru yang berguna untuk proyek Anda, jangan khawatir dan hindari membuat fitur yang tidak berguna.

Langkah 4: Gunakan langkah ini untuk mengubah fitur menjadi skala/unit yang sama. Jika Anda ragu bahwa model yang akan Anda gunakan membutuhkan data dalam skala yang sama, terapkanlah. Itu tidak akan berdampak negatif pada model yang tidak membutuhkan transformasi data.

Langkah 5: Tahap ini menghindari kutukan dimensi, jadi jika Anda merasa mengalami masalah ini, Anda harus menerapkan langkah ini di saluran Anda. Muncul setelah transformasi data karena beberapa teknik (mis., PCA) memerlukan data yang diubah.

Langkah 6: Bagian terakhir sebelum beralih ke fase model adalah menangani data yang tidak seimbang. Selain itu, ada beberapa metrik khusus untuk menghitung performa model saat Anda memiliki masalah ini di data Anda.

Saya tidak menyebutkan langkah pengambilan sampel data di atas, dan alasannya adalah saya mendorong Anda untuk mencoba semua data yang Anda miliki. Jika Anda memiliki data dalam jumlah besar dan tidak dapat menanganinya, pertimbangkan untuk menggunakan pendekatan dari fase pengambilan sampel data.

Penting untuk dicatat bahwa ini mungkin tidak selalu menjadi urutan yang tepat yang harus Anda ikuti, dan Anda mungkin tidak menerapkan semua langkah ini dalam proyek Anda, dan itu akan sepenuhnya bergantung pada masalah dan kumpulan data Anda.

**Kesimpulan**

Fase prapemrosesan data sangat penting untuk menentukan data masukan yang benar untuk algoritme pembelajaran mesin. Seperti yang kita lihat sebelumnya, tanpa menerapkan teknik yang tepat, Anda dapat memperoleh hasil model yang lebih buruk. Misalnya, algoritma k-nearest neighbor dipengaruhi oleh data yang berisik dan redundan, sensitif terhadap skala yang berbeda, dan tidak menangani atribut dalam jumlah besar dengan baik. Jika Anda menggunakan algoritme ini, Anda harus membersihkan data, menghindari dimensi tinggi, dan menormalkan atribut ke skala yang sama.

Namun, jika Anda menggunakan algoritme Pohon Keputusan, Anda tidak perlu khawatir untuk menormalkan atribut ke skala yang sama. Dengan demikian, setiap model memiliki kekhasan masing-masing, dan perlu diketahui terlebih dahulu untuk memberikan input data yang tepat pada model tersebut. Dengan itu, sekarang Anda dapat bergerak maju ke fase eksplorasi model dan mengetahui kekhasan algoritme tersebut.

Satu hal penting terakhir untuk diingat, yang biasanya merupakan kesalahan umum di bidang ini, adalah Anda perlu membagi dataset Anda menjadi set pelatihan dan pengujian sebelum menerapkan beberapa teknik ini, hanya menggunakan set pelatihan untuk mempelajari dan menerapkannya dalam pengujian. bagian. Bagi yang sudah terbiasa dengan Python dan sklearn, Anda menerapkan metode fit and transform pada data pelatihan, dan hanya metode transformasi pada data pengujian.

Daftar Pustaka

<https://www.scalablepath.com/data-science/data-preprocessing-phase>