**PROJEK STATISTIKA INFERENSI**

**Lembar Kerja 6 (LK6)**



Disusun Oleh :

Hafizh Raihan Kurnia Putra (205150207111005)

Ja’far Shidqul Azzam (205150201111006)

Muhammad Rusdy Hanan (205150207111015)

**UNIVERSITAS BRAWIJAYA**

**FAKULTAS ILMU KOMPUTER**

**2023**

**BAB 1**

**PERMASALAHAN & DATASET**

1. **Latar Belakang**

Happy Customer Bank adalah sebuah bank swasta yang bergerak melayani nasabahnya perihal menabung uang, produk investasi, produk kredit, dan sebagainya. Selain layanan tersebut, bank ini juga melakukan cross-selling suatu produk kepada para nasabahnya melalui komunikasi telepon dan jaringan internet. Hal tersebut dilakukan untuk meningkatkan keuntungan yang akan didapatkan oleh bank.

Cross-selling adalah salah satu strategi pemasaran yang digunakan oleh bank untuk menjual layanan tambahan kepada nasabah. Banyak bank yang menggunakan strategi ini untuk meningkatkan pendapatan perusahaan. Contohnya yaitu penawaran kartu kredit dari bank dengan banyak keuntungan yang akan didapat, seperti cashback atau diskon khusus.

Dalam kasus ini, Happy Customer Bank ingin melakukan cross-sell produk kartu kredit mereka kepada para nasabah. Tujuan proyek ini adalah untuk mengetahui nasabah mana yang kemungkinan besar akan tertarik untuk menggunakan layanan kartu kredit. Selain mengetahui tipe nasabah, bank juga akan mengetahui apakah strategi cross-selling ini berjalan dengan baik sesuai dengan tujuan perusahaan untuk mendapat profit pendapatan.

Prediksi dapat menjadi solusi dari permasalahan kesesuaian strategi cross-selling yang telah dilakukan karena dapat membantu bank dalam mengidentifikasi nasabah yang tertarik memiliki kartu kredit. Prediksi dilakukan dengan menganalisis data untuk mengidentifikasi pola nasabah. Dengan mempelajari data tersebut, bank dapat mengembangkan model prediksi untuk mengetahui nasabah yang kemungkinan sangat membutuhkan atau menerima tawaran kartu kredit.

1. **Permasalahan**

Happy Customer Bank ingin melakukan *cross selling* produk kartu kreditnya terhadap nasabah mereka. Untuk alasan efisiensi kinerja *cross selling*, mereka ingin mencari tahu nasabah-nasabah yang memiliki kemungkinan besar untuk membeli produk kartu kredit mereka. Oleh karena itu, kami ingin membuat sebuah mesin yang mampu melakukan prediksi terhadap nasabah yang memiliki kemungkinan besar untuk membeli produk kartu kredit mereka.

1. **Dataset**

Dataset yang akan digunakan berisi data-data mengenai nasabah Happy Customer Bank. Dataset ini memiliki jumlah baris data sebanyak 245725 baris dan kolom sebanyak 10 kolom. Data-data tersebut antara lain adalah ID nasabah pada sistem, jenis kelamin, usia, kode wilayah nasabah, jenis pekerjaan nasabah, kode khusus kanal bank, usia akun bank dalam hari, rata-rata saldo nasabah per tahun, dan status keaktifan produk layanan bank lainnya. Dataset kami dapatkan dari website [www.kaggle.com](http://www.kaggle.com) dengan skor kegunaan sebanyak 8,24 dari 10.

1. **Karakteristik Data**

Berikut adalah karakteristik data yang ada pada setiap fitur/atribut dari dataset.

* Age

Menunjukkan usia pelanggan dengan rentang dari 19 hingga 79 tahun dan bertipe numerik.

* Gender

Bertipe kategorikal dan memiliki dua nilai yang menunjukkan jenis kelamin yaitu ‘Male’ dan ‘Female’ dengan nilai ‘Female’ menjadi nilai yang dominan dalam dataset.

* Region\_code

Merupakan fitur dengan tipe data kategorikal dan memiliki dengan 35 kategori. Data yang sering muncul adalah kategori ‘RG268' dan menunjukkan kode wilayah tempat pelanggan tinggal.

* Occupation

Mencatat jenis pekerjaan pelanggan yang bertipe kategorikal. Terdapat lima kategori pada fitur ini, yaitu 'Self\_Employed', 'Salaried', 'Student', 'Retired', dan 'Others' dengan nilai ‘Self\_Employed’ lebih sering muncul daripada nilai lainnya.

* Channel\_code

Merupakan data dengan tipe kategorikal dengan empat kategori yaitu dari ‘X1’ hingga ‘X4’ yang menunjukkan kode saluran interaksi antara pelanggan dengan bank.

* Vintage

Bertipe numerik yang menunjukkan sudah berapa lama seseorang bergabung dengan bank.

* Credit\_Product

Menunjukkan apakah pelanggan memiliki produk kredit lainnya di bank dengan tipe data kategorikal. Fitur ini memiliki dua kategori yaitu ‘Yes’ atau ‘No’ dengan nilai ‘No’ sebagai modusnya

* Avg\_Account\_Balance

Fitur ini mencatat rata-rata saldo akun pelanggan dalam setahun yang digambarkan dalam tipe data numerik.

* Is\_Active

Menunjukkan apakah pelanggan aktif menggunakan produk bank lainnya. Merupakan fitur dengan tipe data kategorikal dengan dua tipe yaitu ‘Yes’ atau ‘No’ dengan nilai ‘No’ sebagai modus.

* Is\_Lead

Merupakan label yang menunjukkan apakah pelanggan .membeli penawaran kartu kredit. Bertipe data kategorikal dengan dua nilai yaitu 1 untuk membeli dan 0 untuk tidak membeli dengan nilai 0 sebagai modus.

Untuk data bertipe numerik berikut statistik yang lebih lengkap :

| Fitur | Informasi Statistik | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Modus | Mean | Std | Min | Max | Q1 | Q2 | Q3 |
| Age | 28.0 | 43.85 | 14.82 | 23.0 | 85.0 | 30.0 | 43.0 | 54.0 |
| Vintage | 13.0 | 46.95 | 32.35 | 7.0 | 135.0 | 20.0 | 32.0 | 73.0 |
| Avg\_Account\_Balance | 4022185.0 | 1128403 | 852936.4 | 20790 | 10352010 | 604310 | 894601 | 1366666 |

Tabel 1. Informasi statistik fitur bertipe numerik

**BAB 2**

**METODE**

1. **K-Nearest Neighbor**

K-nearest neighbors (KNN) adalah algoritma sederhana namun efektif yang digunakan untuk tugas klasifikasi dan regresi dalam pembelajaran mesin. Ini adalah algoritma non-parametrik yang tidak membuat asumsi tentang distribusi data yang mendasarinya.

Dalam algoritma KNN, dengan diberikan sebuah data baru, K tetangga terdekat dari himpunan data latih diidentifikasi. Jumlah K ditentukan sebelumnya dan dapat disesuaikan sesuai dengan masalah yang sedang diselesaikan. Kemudian, label atau nilai target dari data baru diestimasi berdasarkan mayoritas label atau nilai target dari K tetangga terdekat tersebut. Algoritma ini sangat populer karena mudah dimengerti dan dapat memberikan hasil yang cukup baik pada banyak tugas pembelajaran mesin (Alpaydin, 2020).

1. **Logistic Regression**

Logistic Regression adalah algoritma pembelajaran mesin yang digunakan untuk melakukan klasifikasi pada data dengan dua kelas atau lebih. Dalam logistic regression, variabel dependen atau target (kelas) adalah variabel biner (dua kelas) atau kategori (King & Zeng, 2001). Logistic Regression menghitung probabilitas kelas target untuk setiap data input dengan menggunakan fungsi logit atau sigmoid. Ini menghasilkan nilai-nilai probabilitas yang dapat dikonversi menjadi prediksi kelas dengan mengatur ambang batas probabilitas. Algoritma ini sangat populer karena sederhana, mudah diinterpretasikan, dan dapat digunakan untuk memodelkan banyak masalah klasifikasi yang berbeda.

1. **Gaussian Naive Bayes**

Gaussian Naive Bayes adalah salah satu algoritma *machine learning supervised* yang merupakan variasi dari algoritma Naive Bayes yang mengasumsikan bahwa distribusi fitur pada setiap kelas target adalah Gaussian atau normal. Dalam Gaussian Naive Bayes, setiap fitur input diperlakukan sebagai variabel acak Gaussian yang independen secara kondisional dalam setiap kelas target (Bishop, 2006). Oleh karena itu, model Gaussian Naive Bayes dapat dianggap sebagai model probabilistik yang menggambarkan bagaimana data input tersebar dalam setiap kelas target.

Algoritma ini mempelajari parameter distribusi Gaussian untuk setiap fitur pada setiap kelas target dari data latih. Kemudian, menggunakan Teorema Bayes untuk menghitung probabilitas posteriori untuk setiap kelas target berdasarkan data input. Kelas target dengan probabilitas posteriori tertinggi dianggap sebagai prediksi kelas untuk data input tersebut.

**BAB 3**

**HASIL DAN PEMBAHASAN**

1. **Implementasi**

Proses implementasi dilakukan dengan melakukan preprocessing pada data yang kemudian akan dilakukan pembentukan model disertai dengan hyperparameter tuning untuk mendapatkan performa model yang baik.

1. **Preprocessing**

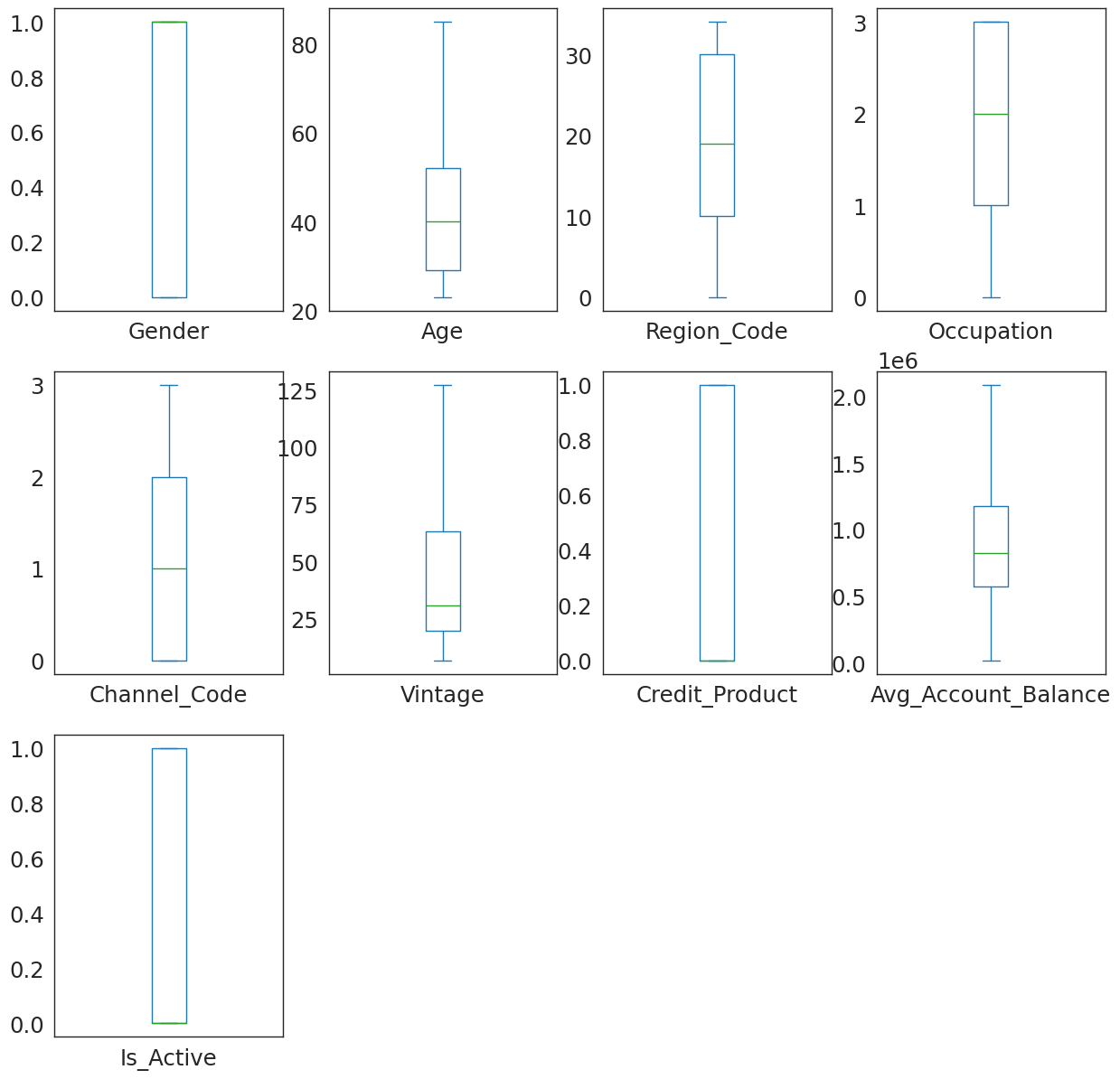
Preprocessing pada data dilakukan dalam beberapa tahap. Yang pertama kali dilakukan dalam proses preprocessing ini adalah mengatasi missing value. Mengatasi missing value dilakukan dengan mencari data yang bernilai NaN.

| Fitur | Hasil Cek Nilai Nan |
| --- | --- |
| ID | False |
| Age | False |
| Region\_Code | False |
| Occupation | False |
| Channel\_Code | False |
| Vintage | False |
| Credit\_Product | True |
| Avg\_Account\_Balance | False |
| Is\_Active | False |
| Is\_Lead | False |

Gambar 1. Tabel Hasil pengecekan missing value pada kolom dataset

Dapat kita ketahui terdapat data yang bernilai NaN yaitu pada kolom Credit\_Product. Untuk mengatasi permasalahan tersebut maka akan dilakukan menghilangkan missing value tersebut dengan melakukan drop pada baris tersebut. Langkah selanjutnya setelah penanganan missing value adalah menghapus nilai yang duplikat hal ini dilakukan dengan fungsi drop duplicate dari pandas. Setelah di drop akan dicek lagi apakah masih terdapat nilai duplikat.

Setelah data duplikat berhasil ditangani maka kita akan melakukan transformasi dari data kategorikal ke nominal dengan menggunakan ordinal encoder. Setelah mengubah data kategorikal maka preprocessing selanjutnya adalah penanganan outlier dengan menggunakan IQR(*inter quartile range).*



Gambar 4. Grafik Box Plot Tiap Fitur Pasca Penanganan Outlier

Penanganan outlier dilakukan untuk menangani pencilan yang terdapat pada data sehingga meningkatkan performa model. Preprocessing yang terakhir adalah melakukan normalisasi dengan menggunakan minmax.

| Fitur | Rentang Data |
| --- | --- |
| ID | 1.0 |
| Age | 1.0 |
| Region\_Code | 1.0 |
| Occupation | 1.0 |
| Channel\_Code | 1.0 |
| Vintage | 1.0 |
| Credit\_Product | 1.0 |
| Avg\_Account\_Balance | 1.0 |
| Is\_Active | 1.0 |
| Is\_Lead | 1.0 |

Gambar 2.Tabel Hasil Normalisasi Data

Normalisasi data dilakukan untuk menyamakan rentang antar fitur, dari gambar diatas dapat dilihat semua fiturnya memiliki rentang yang sama yaitu 1. Hal ini dilakukan untuk meningkatkan performa model yang akan dibentuk.

1. **Hyperparameter Tuning**

Tuning Hyperparameter dilakukan dengan menguji nilai-nilai untuk tiap hyperparameter yang digunakan dalam suatu algoritma. Tuning Hyperparameter akan dilakukan dengan menggunakan library GridsearchCV. GridSearchCV sendiri adalah salah satu teknik tuning parameter dalam machine learning yang digunakan untuk mencari parameter optimal dari suatu model. Pada dasarnya, GridSearchCV adalah sebuah algoritma pencarian parameter terbaik dengan cara melakukan evaluasi performa model pada setiap kombinasi parameter yang didefinisikan di dalam grid. GridsearchCV juga menyediakan cross validation untuk menghindari overfitting pada suatu algoritma. cross validation yang akan digunakan adalah sejumlah 5 split.

1. **Algoritma KNN**

Tuning Hyperparamater pada algoritma KNN( k-Nearest Neighbors) dilakukan dengan menguji 3 hyperparameter dari KNN yaitu Leaf Size, K dan Jenis Jarak dengan memasukkannya dengan nilai yang berbeda beda untuk mendapatkan hasil yang optimal.

Leaf Size adalah parameter dalam algoritma k-Nearest Neighbors (KNN) yang digunakan untuk mengatur ukuran daun (leaf) pada pohon KD-tree yang dibangun untuk menyimpan data. KD-tree adalah struktur data yang digunakan untuk menyimpan data dan memungkinkan kita untuk melakukan pencarian nearest neighbors dengan cepat.

K adalah parameter yang menentukan jumlah tetangga terdekat yang akan dipertimbangkan saat melakukan klasifikasi atau regresi. Dalam k-NN, sebuah titik data akan diklasifikasikan atau diprediksi berdasarkan mayoritas label atau nilai target dari k tetangga terdekatnya dalam dataset.

Jenis Jarak sendiri digunakan untuk menentukan jenis pengukuran jarak yang digunakan. p yang bernilai 1 menunjukkan penggunaan jarak manhattan dan nilai 2 menunjukkan penggunaan jarak euclidean.

Berikut adalah hasil tuning hyperparameter pada algoritma KNN :

| **Hyperparameter** | | | **Akurasi** |
| --- | --- | --- | --- |
| **Leaf Size** | **K** | **Jenis Jarak** |
| 1 | 1 | 1 | 0.798 |
| 1 | 1 | 2 | 0.798 |
| 1 | 3 | 1 | 0.837 |
| 1 | 3 | 2 | 0.837 |
| 1 | 5 | 1 | 0.847 |
| 1 | 5 | 2 | 0.848 |
| 1 | 7 | 1 | 0.852 |
| 1 | 7 | 2 | 0.852 |
| 5 | 1 | 1 | 0.798 |
| 5 | 1 | 2 | 0.798 |
| 5 | 3 | 1 | 0.837 |
| 5 | 3 | 2 | 0.837 |
| 5 | 5 | 1 | 0.847 |
| 5 | 5 | 2 | 0.848 |
| 5 | 7 | 1 | 0.852 |
| 5 | 7 | 2 | 0.852 |
| 10 | 1 | 1 | 0.798 |
| 10 | 1 | 2 | 0.798 |
| 10 | 3 | 1 | 0.837 |
| 10 | 3 | 2 | 0.837 |
| 10 | 5 | 1 | 0.847 |
| 10 | 5 | 2 | 0.848 |
| 10 | 7 | 1 | 0.852 |
| 10 | 7 | 2 | 0.852 |

Tabel 3. Nilai akurasi dari tuning hyperparameter KNN

Berdasarkan pengujian terhadap berbagai hyperparameter yang telah dilakukan, dapat dilihat bahwa hyperparameter untuk mendapatkan hasil yang optimal pada algoritma KNN adalah Leaf Size yang bernilai 1, K yang bernilai 7 dan Jenis Jarak yang bernilai 1 dengan akurasi sebanyak 0.852. Dapat juga kita lihat bahwa hyperparameter yang paling berpengaruh terhadap akurasi adalah n\_neighbour yang menunjukkan jumlah tetangga. Hal ini dilihat dari akurasi yang lebih tinggi untuk tiap kombinasi yang K-nya bernilai 7 dibandingkan kombinasi yang K-nya tidak bernilai 7.

1. **Algoritma Gaussian Naive Bayes**

Tuning Hyperparameter pada algoritma gaussian naive bayes ini dilakukan dengan menguji hyperparameter var\_smoothing dengan memasukkan nilai yang berbeda-beda. Variabel Smoothing adalah parameter dalam algoritma Gaussian Naive Bayes (GNB) yang digunakan untuk mengontrol smoothing level atau regularisasi. Pada dasarnya, var\_smoothing adalah sebuah nilai konstan yang ditambahkan ke setiap variansi fitur agar model tidak overfitting pada data training. Semakin besar nilai var\_smoothing, semakin banyak smoothing yang dilakukan, yang dapat mengurangi overfitting tetapi juga dapat mempengaruhi akurasi model.

| **Variabel Smoothing** | **Akurasi** |
| --- | --- |
| 1 | 0.848 |
| 0.3359818286283782 | 0.846 |
| 0.11288378916846892 | 0.838 |
| 0.0379269019073225 | 0.819 |
| 0.012742749857031341 | 0.811 |
| 0.004281332398719396 | 0.807 |
| 0.0014384498882876629 | 0.806 |
| 0.0004832930238571757 | 0.806 |
| 0.00016237767391887227 | 0.806 |
| 5.455594781168526e-05 | 0.806 |
| 1.8329807108324375e-05 | 0.805 |
| 6.158482110660267e-06 | 0.805 |
| 2.06913808111479e-06 | 0.805 |
| 6.951927961775605e-07 | 0.805 |
| 2.3357214690901262e-07 | 0.805 |
| 7.847599703514623e-08 | 0.805 |
| 2.6366508987303608e-08 | 0.805 |
| 8.858667904100832e-09 | 0.805 |
| 2.9763514416313252e-09 | 0.805 |

Tabel 4. Nilai akurasi dari tuning hyperparameter GNB

Berdasarkan pengujian terhadap berbagai hyperparameter yang telah dilakukan, dapat dilihat bahwa hyperparameter untuk mendapatkan hasil yang optimal pada algoritma Gaussian Naive Bayes adalah Variabel Smoothing yang bernilai 1 yang hasilnya memiliki akurasi sebanyak 0.848.

1. **Algoritma Logistic Regression**

Pada Algoritma Logistic Regression hyperparameter yang akan di-tuning adalah Regularisasi, Iterasi Maksimal dan Algoritma Optimasi. Regularisasi adalah parameter dalam algoritma Logistic Regression yang digunakan untuk mengontrol regularisasi model. Semakin kecil nilai Regularisasi, semakin banyak regularisasi yang dilakukan, dan semakin besar nilai Regularisasi, semakin kecil regularisasi yang dilakukan. Artinya, semakin besar nilai Regularisasi, semakin kompleks model logistic regression dan semakin kecil nilai Regularisasi, semakin simpel model logistic regression.

Algoritma Optimasi adalah parameter dalam algoritma Logistic Regression yang digunakan untuk menentukan algoritma solver yang digunakan untuk mengoptimasi parameter model, terdapat beberapa pilihan seperti newton-cg, lbfgs, sag, saga, dan lainnya.

Sedangkan Iterasi Maksimal sendiri adalah parameter dalam algoritma Logistic Regression yang digunakan untuk menentukan jumlah maksimum iterasi yang diperbolehkan untuk mencapai konvergensi. Iterasi akan terus dilakukan sampai model mencapai konvergensi atau jumlah maksimum iterasi yang diperbolehkan telah dicapai.

| **Hyperparameter** | | | **Akurasi** |
| --- | --- | --- | --- |
| **Regularisasi** | **Iterasi Maksimal** | **Algoritma Optimasi** |
| 0.1 | 100 | newton-cg | 0.848 |
| 0.1 | 100 | lbfgs | 0.848 |
| 0.1 | 100 | liblinear | 0.848 |
| 0.1 | 200 | newton-cg | 0.848 |
| 0.1 | 200 | lbfgs | 0.848 |
| 0.1 | 200 | liblinear | 0.848 |
| 0.1 | 400 | newton-cg | 0.848 |
| 0.1 | 400 | lbfgs | 0.848 |
| 0.1 | 400 | liblinear | 0.848 |
| 1 | 100 | newton-cg | 0.848 |
| 1 | 100 | lbfgs | 0.848 |
| 1 | 100 | liblinear | 0.848 |
| 1 | 200 | newton-cg | 0.848 |
| 1 | 200 | lbfgs | 0.848 |
| 1 | 200 | liblinear | 0.848 |
| 1 | 400 | newton-cg | 0.848 |
| 1 | 400 | lbfgs | 0.848 |
| 1 | 400 | liblinear | 0.848 |
| 10 | 100 | newton-cg | 0.848 |
| 10 | 100 | lbfgs | 0.848 |
| 10 | 100 | liblinear | 0.848 |
| 10 | 200 | newton-cg | 0.848 |
| 10 | 200 | lbfgs | 0.848 |
| 10 | 200 | liblinear | 0.848 |
| 10 | 400 | newton-cg | 0.848 |
| 10 | 400 | lbfgs | 0.848 |
| 10 | 400 | liblinear | 0.848 |

Tabel 5. Nilai akurasi dari tuning hyperparameter Logistic Regression

Hasil dari tuning hyperparameter tidak signifikan dari variabel yang ada. Hal ini dapat dilihat dari nilai akurasi yang sama untuk tiap kombinasi dari hyperparameter yang berbeda-beda. Walau begitu, hasil dari Grid Search CV menyatakan bahwa parameter terbaiknya yaitu Regularisasi dengan nilai 0.1, Iterasi Maksimal dengan nilai 100, dan Algoritma Optimasi yaitu liblinear.

1. **Hasil Pengujian**

Pengujian model dengan data uji perlu dilakukan untuk menguji data yang belum pernah digunakan sehingga dapat dilihat kemampuan model dalam memprediksi data yang baru. Data uji sendiri memiliki proporsi dengan data latih yaitu 20 : 80. Model yang dilatih dengan data latih dengan menggunakan hyperparameter yang telah dipilih sebelumnya dari masing-masing metode untuk mendapatkan hasil yang optimal untuk setiap algoritma.

1. **Algoritma KNN**

Hasil Pengujian dengan menggunakan algoritma KNN dengan Hyperparameter berupa K bernilai 7, Leaf Size bernilai 1, dan Jenis Jarak bernilai 1 mendapatkan hasil sebagai berikut

| Label | Precision | Recall | F1-score |
| --- | --- | --- | --- |
| 0 | 0.88 | 0.96 | 0.92 |
| 1 | 0.53 | 0.25 | 0.34 |
| Akurasi | | | 0.85 |

Tabel 6. Hasil pengujian dengan Algoritma KNN

| Matriks Konfusi | | **Prediksi** | |
| --- | --- | --- | --- |
| **0** | **1** |
| **Aktual** | **0** | 33061 | 1413 |
| **1** | 4623 | 1566 |

Tabel 7. Matriks Konfusi Algoritma KNN

Dapat dilihat dari confusion matrix bahwa data yang berlabel 0 dan terprediksi benar sejumlah 33061 dan yang terprediksi salah sejumlah 1566. Untuk data yang berlabel 1 sendiri terdapat sejumlah 4623 yang terprediksi salah dan 1566 yang terprediksi benar .

Untuk label 0 nilai presisinya adalah 0.88 dengan recall 0.96, dan f1-score senilai 0.92. Untuk label 1, nilai presisinya sendiri adalah 0.53 dengan recall 0.25 dan f1-score 0.34. Akurasi yang dihasilkan oleh model ini sebesar 85%..

1. **Algoritma Gaussian Naive Bayes**

Hasil Pengujian dengan menggunakan algoritma Gaussian Naive Bayes dengan Hyperparameter Variabel Smoothing bernilai 1.

| Label | Precision | Recall | F1-score |
| --- | --- | --- | --- |
| 0 | 0.85 | 1.00 | 0.92 |
| 1 | 0.00 | 0.00 | 0.00 |
| Akurasi | | | 0.85 |

Tabel 8. Hasil pengujian dengan Algoritma Naive Bayes

| Matriks Konfusi | | **Prediksi** | |
| --- | --- | --- | --- |
| **0** | **1** |
| **Aktual** | **0** | 34474 | 0 |
| **1** | 6189 | 0 |

Tabel 9. Matriks Konfusi Gaussian Naive Bayes

Dapat dilihat dari confusion matrix bahwa data yang berlabel 0 dan terprediksi benar sejumlah 31716 dan yang terprediksi salah sejumlah 2758. Untuk data yang berlabel 1 sendiri terdapat sejumlah 6189 yang terprediksi salah dan 0 yang terprediksi benar. Terdapat error saat proses perhitungan yang mengakibatkan pembagian bilangan dengan 0.

Untuk label 0 nilai presisinya adalah 0.85 dengan recall 1.00, dan f1-score senilai 0.92. Untuk label 1, nilai presisinya sendiri adalah 0.00 dengan recall 0.00 dan f1-score 0.00. Akurasi Model yang didapatkan adalah 0.85.

1. **Algoritma Logistic Regression**

Hasil Pengujian dengan menggunakan algoritma Logistic Regression dengan Hyperparameter Regularisasi dengan nilai 0.1, Iterasi Maksimal dengan nilai 100, dan Algoritma Optimasi yaitu liblinear.

| Label | Precision | Recall | F1-score |
| --- | --- | --- | --- |
| 0 | 0.86 | 0.98 | 0.92 |
| 1 | 0.49 | 0.09 | 0.15 |
| Akurasi | | | 0.85 |

Tabel 10. Hasil pengujian dengan Algoritma Logistic Regression

| Matriks Konfusi | | **Prediksi** | |
| --- | --- | --- | --- |
| **0** | **1** |
| **Aktual** | **0** | 33892 | 582 |
| **1** | 5625 | 564 |

Tabel 11. Matriks Konfusi Logistic Regression

Dapat dilihat dari confusion matrix bahwa data yang berlabel 0 dan terprediksi benar sejumlah 33892 dan yang terprediksi salah sejumlah 582 Untuk data yang berlabel 1 sendiri terdapat sejumlah 5625 yang terprediksi salah dan 564 yang terprediksi benar.

Untuk label 0 nilai presisinya adalah 0.86 dengan recall 0.98, dan f1-score senilai 0.92. Untuk label 1, nilai presisinya sendiri adalah 0.49 dengan recall 0.09 dan f1-score 0.15. Akurasi yang dihasilkan oleh model ini sebesar 85%.

1. **Analisis dan Pembahasan**
2. **ANOVA (Analisis Variansi)**

Berdasarkan hasil pengujian menggunakan model KNN, GNB, dan Logisitic Regression di atas, dapat terlihat bahwa ketiga model tersebut memiliki nilai akurasi F1-score yang sama yaitu 0.85. Nilai tersebut dianggap cukup tinggi, tetapi nilai akurasi F1-score untuk data yang berlabel 1 memiliki nilai yang rendah pada ketiga model tersebut dan yang berlabel 0 memiliki nilai akurasi F1-score yang sangat tinggi pada ketiga model. Hal ini kemungkinan terjadi karena kurangnya data latih untuk data yang memiliki label 1 (*underfitting*) mengingat modus data pada fitur Is\_Lead adalah data yang memiliki label 0.

Kami juga melakukan uji anova satu arah karena ketiga model tersebut terlihat memiliki kesamaan pada nilai akurasi F1-score. Kami ingin melihat apakah terdapat benar terdapat kesamaan pada nilai akurasi F1-score ketiga model tersebut. Berikut adalah data hasil akurasi dari metode KNN, Gaussian Naive Bayes (GNB), dan Logistic Regression.

| **KNN** | **GNB** | **Logistic Regression** |
| --- | --- | --- |
| 0.851 | 0.845 | 0.846 |
| 0.850 | 0.846 | 0.847 |
| 0.850 | 0.848 | 0.848 |
| 0.851 | 0.847 | 0.847 |
| 0.855 | 0.850 | 0.851 |

Tabel 6. Hasil akurasi 5-Fold dengan KNN, GNB, dan Logistic Regression

Setelah menggunakan fungsi **f\_oneway()** dari library statsmodel maka didapatkan nilai-p sebesar 0.00996 dan F sebesar 6.93408.

| **stat** | **df** | **sum\_sq** | **mean\_sq** | **F** | **PR(>F)** |
| --- | --- | --- | --- | --- | --- |
| model | 2.0 | 0.000052 | 0.000026 | 6.934087 | 0.009965 |
| Residual | 12.0 | 0.000045 | 0.000004 | NaN | NaN |

Tabel 7. ANOVA 1 arah menggunakan library bioinfokit

Nilai-p 0.00996 < 0.05 sehingga H0 ditolak. Hal tersebut berarti bahwa terdapat perbedaan rata-rata pada hasil kinerja akurasi dari ketiga algoritma tersebut.

**3.3.2 Analisis Pasca Anova**

Setelah penolakan H0, kita akan mencari metode-metode yang memiliki perbedaan rata-rata yang jauh menggunakan tukey-kramer.

| **group1** | **group2** | **meandiff** | **p-adj** | **lower** | **upper** | **reject** |
| --- | --- | --- | --- | --- | --- | --- |
| gnb | knn | 0.0041 | 0.0153 | 0.0008 | 0.0073 | True |
| gnb | logreg | 0.0003 | 0.9725 | -0.0030 | 0.0035 | False |
| knn | logreg | -0.0038 | 0.0229 | -0.0071 | -0.0005 | True |

Tabel 8. Hasil analisis tukey-kramer

Berdasarkan hasil tabel tersebut, kita dapat melihat bahwa terdapat perbedaan secara signifikan antara model GNB dan KNN karena memiliki nilai-p 0.0153 (lebih rendah dari nilai alpha 0.05). Selain itu, perbedaan secara signifikan juga dapat terlihat pada model KNN dan Logistic Regression karena memiliki nilai-p 0.0229 (lebih rendah dari nilai alpha 0.05).

**DAFTAR PUSTAKA**

Garg, Shelvi. 2022. Credit Card Lead Prediction. URL : <https://www.kaggle.com/datasets/shelvigarg/credit-card-buyers> diakses pada 12 Maret 2023

SEPAKAT. 2018. Kelompok Usia. URL : [Kelompok Usia - SEPAKAT wiki (bappenas.go.id)](https://sepakat.bappenas.go.id/wiki/Kelompok_Usia) diakses pada 19 Maret 2023

Ansari, M. kaish. (2020, November 22). How to connect Google Colab with Google Drive. MarkTechPost. Retrieved March 20, 2023, from <https://www.marktechpost.com/2019/06/07/how-to-connect-google-colab-with-google-drive/>

Alpaydin, E. (2010). Introduction to Machine Learning (2nd ed.). Cambridge, MA: MIT Press.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. New York: Springer.

King, G., & Zeng, L. (2001). Logistic Regression in Rare Events Data. Political Analysis, 9(2), 137–163.