**TAHAP DASAR**

1. **Dataset**

Dalam tugas ini mengangkat suatu studi kasus klasifikasi penyakit liver dengan menggunakan dataset Penderita Penyakit Liver di India atau ILPD (Indian Liver Patient Dataset) yang bisa diakses pada repoistory UCI Machine Learning Repository: <https://archive.ics.uci.edu/ml/datasets/ILPD+(Indian+Liver+Patient+Dataset)>.

* Metadata:

Dataset Characteristic : Multivariate

Attribute Characteristic : Integer, Real

Associated Tasks : Classification

Number of Instance : 583

Number of Atribute : 10

Missing Values? : N/A

Area : Life

Date Donated : 2012-05-21

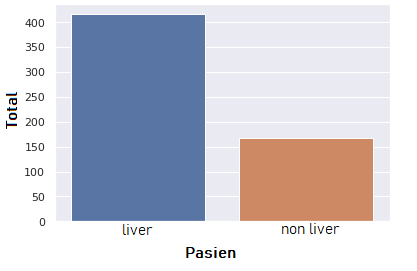
Number of Web Hits : 176804

* Fitur Dataset:

|  |  |  |
| --- | --- | --- |
| **No.** | **Fitur** | **Type** |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11. | age  gender  top\_bilirubin  direct\_bilirubin  tot\_proteins  albumin  ag\_ratio  sgpt  sgot  alkphos  is\_patient | int64  object  float64  float64  int64  int64  int64  float64  float64  float64  int64 |

* Distribusi Dataset

Dataset ini memiliki 2 target kelas yakni pada fitur is\_patient dengan isi kelas liver sebanyak 441 dan non liver 142. Dataset ini tergolong dataset yang imbalance sehingga memerlukan teknik balancing dataset dengan menerpakan oversampling sebagai metode balancing data. Tabel distribusi dapat dilihat pada gambar dibawah



Gambar 1. Distribusi Dataset

1. **Algoritma Machine Learning Yang Digunakan**

Pada tugas kali ini menggunakan algoritma machine learning Random Forest yang dimana metode klasifikasi random forest adalah metode pohon gabungan pengembangan metode Classification and Regression Tree (CART) dengan menerapkan metode bootstrap aggregating (bagging) dan random feature selection. Tujuannya digunakan untuk pengklasifikasian data set dalam jumlah besar. Karena fungsinya bisa digunakan untuk banyak dimensi dengan berbagai skala dan performa yang tinggi. Pada tugas ini menerapkan splitting dataset atau pembagian persebaran data. Tujuannya splitting dataset dengan data latih dengan data uji karena untuk menghindari terjadinya overfitting, yaitu suatu kondisi pelatihan yang hasil uji terhadap data yang dilatih sangat bagus tetapi diuji oleh data lain yang tidak digunakan dalam pelatihan sangat buruk.

1. **Uji Peforma**

Dalam penugasan ini menerapkan beberapa matriks pengujian yakni:

1. Accuracy:

Merupakan rasio prediksi Benar (positif dan negatif) dengan keseluruhan data. Akurasi dapat diperoleh dengan menggunakan rumus

1. Precision:

Merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positf. Precision dapat diperoleh dengan menggunakan rumus

1. Recall:

Merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Recall dapat diperoleh dengan menggunakan rumus

1. F1-Score:   
   F1 Score merupakan perbandingan rata-rata presisi dan recall yang dibobotkan. F1 Score dapat diperoleh dengan menggunakan rumus
2. ROC-AUC:   
   ROC (Receiver Operating Characteristics) adalah semacam alat ukur performance untuk classification problem dalam menentukan threshold dari suatu model. AUC adalah luas area di bawah curve ROC, atau integral dari fungsi ROC.
3. Fit Time:

Waktu pemasangan estimator pada train set untuk setiap cv split.

1. Score Time:

Waktu untuk menilai estimator pada set tes untuk setiap pembagian cv.

1. Test Score:

Susunan skor untuk nilai tes pada setiap pembagian cv.

**4. Cross Validation**

Penugasan ini menerapakan Cross Validation dengan tujuan untuk mengukur uji peforma parameter yang dipilih untuk model yang dibuat. Metode yang dipilih untuk Cross Validation adalah dengan menerapkan K-Fold Validation dengan besaran dikarenakan 10 fold CV adalah salah satu dari K fold CV yang disarankan untuk pemilihan model terbaik karena cenderung memberikan estimasi akurasi yang tidak terlalu bias jika dibandingkan dengan CV biasa, leave-one-out CV, dan bootstrap. Hasil dari validasi peforma yang telah dibuat dapat dilihat dalam tabel berikut ini:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Classification Report** | | | | | | |
| **Algoritma** | | **Accuracy** | **Precission** | **Recall** | **F1-Score** | **Roc\_auc** |
| Random Forest | Train Perf | 1.00 | 1.00 | 1.00 | 1.00 | 0.57 |
| Test Perf | 0.68 | 0.59 | 0.56 | 0.55 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Cross Validation K=10** | | | |
| **Algoritma** | **Fit Time** | **Score Time** | **Test Score** |
| Random  Forest | [0.28908753 0.23939109 0.30658817 0.44484973 0.50295401 0.15020847  0.15259552 0.15171123 0.15959072 0.15307236] | [0.0392549 0.02651429 0.04057765 0.07869506 0.02539015 0.02387762  0.02260327 0.02415252 0.02491808 0.02280164] | [0.78723404 0.65957447 0.61702128 0.78723404 0.80851064 0.5106383  0.67391304 0.63043478 0.67391304 0.86956522] |

**5. Algoritma Machine Learning Lain**

Pada Penugasan ini menggunakan beberapa Algoritma Machine Learning yakni:

1. **SVM**

SVM adalah teknik supervised classification yang cukup rumit tetapi memiliki tingkat keakuratan yang cukup baik. Algoritma Support Vector Machine akan menemukan hyperplane atau batas antara dua kelas atau lebih dengan memaksimalkan margin di antara kelas-kelas tersebut. Tujuannya SVM digunakan untuk mencari hyperplane terbaik dengan memaksimalkan jarak antar kelas.

1. **K-Neighbors**

Algoritma KNN adalah algoritma generalisasi untuk terdekat peraturan tetangga. Offset induktifnya adalah label kelas dari k-sample dengan label kelas yang akan diuji paling mirip dengan yang terdekat. Tujuan dari algoritma KNN adalah untuk mengklasifikasi objek baru berdasarkan atribut dan training samples. Dimana hasil dari sampel uji yang baru diklasifikasikan berdasarkan mayoritas dari kategori pada KNN. Jarak yang digunakan adalah jarak Euclidean Distance.

1. **XGBoost**

Algoritma XGBoost merupakan salah satu algoritma yang paling populer dan paling banyak digunakan karena algoritma ini termasuk algoritma yang powerful. Pada dasarnya, algoritma ini sama dengan algoritma gradient boost hanya saja menggunakan beberapa proses tambahan sehingga lebih powerful.

1. **EXTree**

Klasifikasi dengan metode extra trees atau yang disebut juga sebagai Extremly randomized Trees merupakan varian pengembangan dari decision tree acak pada berbagai sub bagian dataset dan menghitung rata-ratanya untuk meningkatkan akurasi prediksi dan pengendalian overvitting.

1. **ANN**

Artificial Neural Network (ANN) atau jaringan syaraf tiruan adalah jaringan dari sekelompok unit pemroses kecil yang dimodelkan berdasarkan perilaku jaringan syaraf manusia. ANN sendiri memiliki 3 jenis layer untuk bekerja yakni input, hidden, dan output layer. ANN disini merupakaan salah satu solusi penujian yang potensial dari metode lain diatas. Berikut perbandingan 5 Algoritma di atas dapat dilihat dalam tabel dibawah ini:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Classification Report** | | | | | | |
| **Algoritma** | | **Accuracy** | **Precission** | **Recall** | **F1-Score** | **Roc\_auc** |
| SVM | Train Perf | 0.72 | 0.36 | 0.50 | 0.42 | 0.50 |
| Test Perf | 0.68 | 0.34 | 0.50 | 0.41 |
| K-Neighbors | Train Perf | 0.77 | 0.71 | 0.67 | 0.69 | 0.58 |
| Test Perf | 0.72 | 0.67 | 0.61 | 0.62 |
| XGBoost | Train Perf | 0.92 | 0.93 | 0.87 | 0.89 | 0.59 |
| Test Perf | 0.68 | 0.62 | 0.59 | 0.59 |
| EXTree | Train Perf | 1.00 | 1.00 | 1.00 | 1.00 | 0.60 |
| Test Perf | 0.74 | 0.74 | 0.62 | 0.63 |
| ANN | Train Perf | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Test Perf | 1.00 | 1.00 | 1.00 | 1.00 |

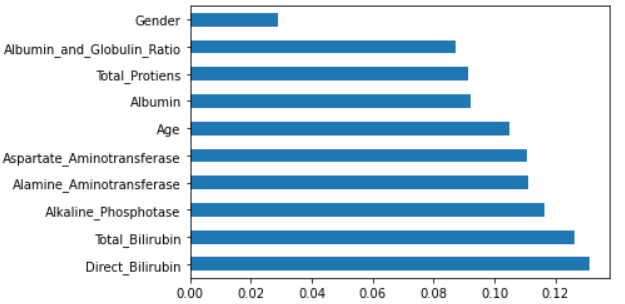
|  |  |  |  |
| --- | --- | --- | --- |
| **Cross Validation K=10** | | | |
| **Algoritma** | **Fit Time** | **Score Time** | **Test Score** |
| SVM | [0.01174974 0.00716972 0.00782132 0.00768876 0.00763178 0.00746989  0.00764632  0.0077765 0.00775361 0.00770164] | [0.00205851 0.00199366 0.00197697 0.00193834 0.0021615 0.00189638  0.00194192 0.0019021 0.00194764 0.00198197] | [0.78723404 0.70212766 0.63829787 0.87234043 0.80851064 0.53191489  0.7826087 0.60869565 0.63043478 0.84782609] |
| K-Neighbors | [0.00600123 0.00209451 0.00199485 0.00215173 0.00191379 0.00181413  0.0017252 0.00189209 0.00192904 0.00188017] | [0.00447631 0.00296021 0.00291419 0.00308609 0.00289345 0.00327563  0.00260687 0.00291944 0.00281382 0.00295877] | [0.74468085 0.65957447 0.65957447 0.74468085 0.68085106 0.57446809  0.67391304 0.56521739 0.63043478 0.7826087 ] |
| XGBoost | [1.16964316 0.0345943 0.0289526 0.02854609 0.02892613 0.02897644  0.02984977 0.03167534 0.0291636 0.02898145] | [0.0015738 0.00109053 0.00108051 0.00108695 0.0010736 0.0009861  0.00118494 0.00103354 0.00102162 0.00101542] | [0.68085106 0.63829787 0.57446809 0.70212766 0.78723404 0.55319149  0.76086957 0.69565217 0.58695652 0.82608696] |
| EXTRee | [0.34243584 0.17325234 0.10066891 0.10791206 0.12656116 0.12586069  0.12274694 0.1331656 0.12333989 0.12036872] | [0.02075028 0.01034307  0.0108254 0.01030993 0.01151896 0.01155663  0.01130176 0.0114615 0.01301694 0.01058936] | [0.78723404 0.65957447 0.68085106 0.85106383 0.85106383 0.55319149  0.7826087 0.69565217 0.63043478 0.82608696] |

1. **Feature Selection**

Pada tahap ini diberikan beberapa pendekatan metode feature selection yang akan digunakan. Dimana pada skenario test dalam metode feature selection data sudah diberikan pendekatan oversampling dan juga diberikan model SVM sebagai pilihan utama model pelatihan. Berikut metode feature selectionn yang digunakan:

1. Feature Importance

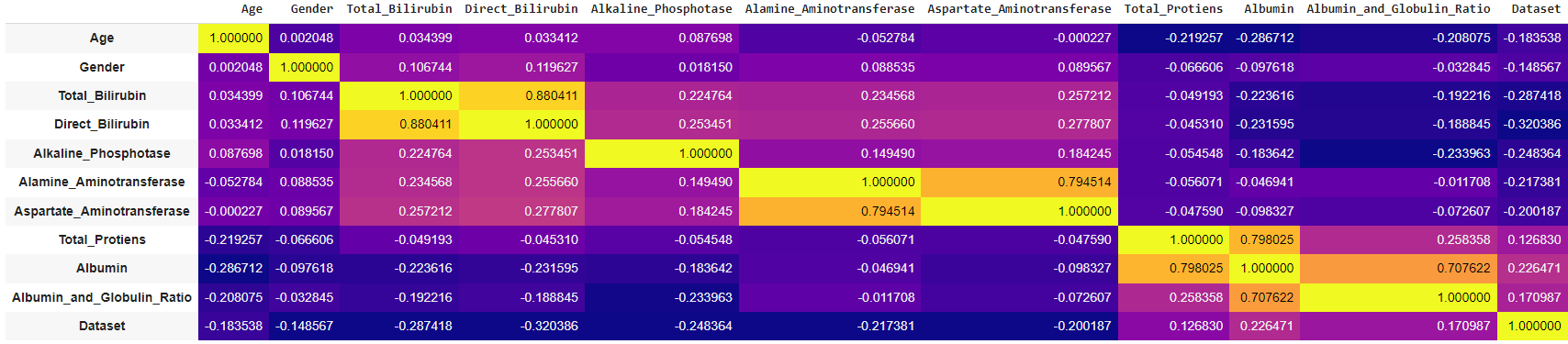
Dalam skenario feature importance ini akan menampilkan sesi hasil uji coba feature selection yang dimana metode pertama menggunakan feature importance untuk langsung dapat menentukan hasil importance dari setiap feature dapat dilihat di Gambar dibawah.



Gambar 2. Gambar Feature Importance

1. Correlation Matrix dengan menggunakan Heatmap

Tahap selanjutnya adalah teknik feature correlation matrix dengan menggunakan heatmap yang mendapati hasil dari gambar dibawah, terdapat beberapa fitur yang memiliki korelasi yang kuat yakni:

* direct\_bilirubin & total\_bilirubin
* aspartate\_aminotransferase & alamine\_aminotransferase
* total\_protiens & albumin
* albumin\_and\_globulin\_ratio & albumin

Gambar 3. Correlation Heatmap

Dengan didapatkannya hasil feature correlation tersebut diterakpan teknik pengurangan jumlah fitur yang terlibat dalam menentukan suatu nilai kelas target. Fitur yang diabaikan biasanya berupa fitur yang tidak relevan dan data berlebih. Tujuan utama dari seleksi fitur ialah memilih fitur terbaik dari suatu kumpulan data fitur. Berikut ini adalah Feature yang diseleksi:

* age
* gender
* total\_bilirubin
* alkaline\_phosphotase
* alamine\_aminotransferase
* albumin\_and\_globulin\_ratio
* is\_patient

Dengan hasil uji performa dari 7 feature selection diatas dengan menggunakan model SVM memiliki hasil akurasi 0.87 sebesar dan Roc\_auc sebsar 0.91.

1. Test Per Feature

Dalam teknik ini akan menggunakan model SVM dengan direktori train akan dipilih dari hasil uji peforma pada setiap feature dimulai dari pelatihan 1 feature sampai 10 feature dengan hasil uji terbaik dari feature nya adalah sebagai berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jumlah Feature** | **Feature Terbaik** | **Hasil** | |
| **Akurasi** | **Roc\_auc** |
| 1 Feature | Alkaline\_Phosphotase | 0.66 | 0.665 |
| 2 Feature | Alkaline\_Phosphotase dan Age | 0.68 | 0.683 |
| 3 Feature | Alkaline\_Phosphotase, Age, dan Albumin | 0.72 | 0.725 |
| 4 Feature | Alkaline\_Phosphotase, Age, dan Albumin, dan Aspartate\_Aminotransferase | 0.83 | 0.826 |
| Alkaline\_Phosphotase, Age, dan Albumin, dan Albumin\_and\_Globulin\_Ratio | 0.83 | 0.826 |
| 6 Feature | Alkaline\_Phosphotase, Age, Albumin, Aspartate\_Aminotransferase, dan Gender | 0.82 | 0.821 |
| 7 Feature | Alkaline\_Phosphotase','Age','Albumin','Aspartate\_Aminotransferase','Gender','Total\_Protiens' | 0.83 | 0.827 |

Ada beberapa fitur yang tidak dimasukkan yakni feature Albumin\_and\_Globulin\_Ratio

Dikarenakan memiliki hasil uji akurasi dan Roc\_auc yang sama dengan Alkaline\_Phosphotase sehingga dihilangkan atau tidak digunakan hal ini dapat dilihat pada Jumlah 5 Feature. Untuk 3 feature lain yakni:

* Direct\_Bilirubin
* Alamine\_Aminotransferase
* Total\_Proteins

Tidak dimasukkan ke dalam pelatihan dikarenakan ketika pelatihan pada jumlah 7 feature ketiga feature tersebut tidak meningkatkan akurasi sehingga tidak digunakan.

Sehingga hasil dari ketiga metode feature selection diatas dapat disimpulkan untuk menggunakan hasil dari teknik kedua yakni Correlation Matrix dengan menggunakan Heatmap dikarenakan mendapati hasil terbaik pada uji peforma.

1. **Hyperparameter Tuning**

Grid Search adalah salah satu metode untuk mengetahui kombinasi hyperparameter. Mean Cross Validation berfungsi sebagai metrik pengukuran kinerja Grid Search (CV). Nilai-nilai yang kita masukkan untuk Hyperparameter digabungkan sebagai bagian dari proses Grid Search. Misalnya, Grid Search akan mencari semua kemungkinan kombinasi hyperparameter A dan B, yaitu dan memilih kombinasi terbaik berdasarkan nilai CV Score tertinggi. Tabel di bawah ini adalah hasil parameter terbaik algoritma yang digunakan berdasarkan nilai CV Score tertinggi.

|  |  |  |
| --- | --- | --- |
| **Algoritma** | **Hyperparameter** | **CV Score** |
| SVM | {'C': 100, 'gamma': 1, 'kernel': 'rbf'} | 0.85 |
| K-Neighbors | {'leaf\_size': 1, 'n\_neighbors': 2, 'p': 1} | 0.84 |
| Random Forest | {'criterion': 'entropy', 'max\_leaf\_nodes': None, 'min\_samples\_leaf': 1, 'min\_samples\_split': 2, 'n\_estimators': 500} | 0.95 |
| XGBoost | {'criterion': 'gini', 'max\_leaf\_nodes': 4, 'min\_samples\_leaf': 1, 'min\_samples\_split': 1, 'n\_estimators': 500} | 0.90 |
| EXTree | {'criterion': 'gini', 'max\_leaf\_nodes': None, 'min\_samples\_leaf': 1, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.97 |

1. **Hasil Classification Report dengan Feature Selection dan Parameter Tuning**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Classification Report** | | | | | | |
| **Algoritma** | | **Accuracy** | **Precission** | **Recall** | **F1-Score** | **Roc\_auc** |
| SVM | Train Perf | 0.99 | 0.99 | 0.99 | 0.99 | 0.87 |
| Test Perf | 0.87 | 0.87 | 0.87 | 0.87 |
| K-Neighbors | Train Perf | 0.96 | 0.96 | 0.96 | 0.96 | 0.78 |
| Test Perf | 0.78 | 0.78 | 0.78 | 0.78 |
| XGBoost | Train Perf | 1.00 | 1.00 | 1.00 | 1.00 | 0.83 |
| Test Perf | 0.83 | 0.84 | 0.84 | 0.83 |
| EXTree | Train Perf | 1.00 | 1.00 | 1.00 | 1.00 | 0.90 |
| Test Perf | 0.89 | 0.90 | 0.90 | 0.89 |
| Random Forest | Train Perf | 1.00 | 1.00 | 1.00 | 1.00 | 0.85 |
| Test Perf | 0.83 | 0.85 | 0.85 | 0.83 |

1. **Hasil Cross Validation Dengan Feature Selection dan Parameter Tuning**

|  |  |  |  |
| --- | --- | --- | --- |
| **Cross Validation K=10** | | | |
| **Algoritma** | **Fit Time** | **Score Time** | **Test Score** |
| SVM | [0.2917316 0.31931424 0.28942776 0.29250503 0.28609991 0.34061885  0.27373385 0.35501456 0.38555336 0.26865101] | [0.00556231 0.01963115 0.00582671 0.00590539 0.00565863 0.0057056  0.00543904 0.0136168 0.00811982 0.00553513] | [0.82089552 0.82089552 0.8358209 0.91044776 0.80597015 0.87878788  0.77272727 0.86363636 0.86363636 0.84848485] |
| K-Neighbors | [0.01605606 0.01644373 0.0147593 0.02115035 0.00799608 0.01447725  0.01619577 0.01749754 0.01479292 0.01450515] | [0.03687692 0.01794934 0.03372335 0.03938746 0.04467702 0.01612401  0.04728866 0.02795196 0.03407431 0.01849079] | [0.80597015 0.82089552 0.80597015 0.8358209 0.71641791 0.84848485  0.84848485 0.84848485 0.74242424 0.74242424] |
| XGBoost | [0.06385922 0.06261301 0.06657743 0.06562614 0.0672164 0.06349754  0.06666374 0.06121063 0.06810403 0.06173396] | [0.00212836 0.00186443 0.00183344 0.0025146 0.00191569 0.00188303  0.00187874 0.00188136 0.0019877 0.00186801] | [0.8358209 0.80597015 0.8358209 0.88059701 0.88059701 0.86363636  0.83333333 0.89393939 0.86363636 0.78787879] |
| EXTree | [0.26514721 0.25632811 0.26407695 0.26392579 0.2476933 0.24653673  0.24409342 0.24945951 0.26823401 0.25323319] | [0.02344298 0.02529097 0.02655077 0.02351499 0.0231092 0.03202319  0.02370572 0.02376103 0.02315879 0.02327967] | [0.85074627 0.88059701 0.88059701 0.92537313 0.8358209 0.87878788  0.84848485 0.90909091 0.75757576 0.86363636] |
| Random Forest | [1.07697296 1.03687549 1.13470769 1.79016042 1.71760297 1.23317933  1.02376223 1.01432776 1.03341651 1.00061345] | [0.05342531 0.05623698 0.10266423 0.08170557 0.08432221 0.05213475  0.05163217 0.05963302 0.05185986 0.05156446] | [0.82089552 0.80597015 0.86567164 0.86567164 0.85074627 0.84848485  0.8030303 0.87878788 0.83333333 0.84848485] |