**CMAPSS Jet Engine Failure Classification Based On Sensor Data**

1. **Overview**

Badan antariksa Amerika Serikat atau yang populer dikenal sebagai NASA beberapa waktu lalu membagikan dataset yang berisi data simulasi mesin jet. Data ini berisi data sensor dari suatu mesin jet mulai dari mesin pertama kali digunakan sampai mesin tersebut mati. Hal ini tentunya menarik untuk dibahas bagaimana cara kita bisa mengenali pola data sensor kemudian melakukan klasifikasi untuk menentukan suatu mesin jet masih berfungsi normal atau sudah gagal. Klasifikasi merupakan salah satu kemampuan dari machine learning yang dapat memprediksi suatu kejadian berdasarkan data historis. Output dari klasifikasi ini biasanya berupa data kategorikal. Salah satu contohnya yaitu klasifikasi kegagalan mesin jet berdasarkan data sensor. Proyek ini akan membahas bagaimana model machine learning belajar agar dapat digunakan untuk memprediksi kesehatan mesin. Proyek ini dikerjakan dengan mengacu pada konsep CRISP-DM yaitu sebuah workflow yang mengatur proses data mining. Untuk lebih lengkapnya, mari kita lihat bersama-sama.

1. **Table of Content**
2. Business Understanding

* Mengapa prediksi kerusakan mesin penting?
* Apa masalahnya?
* Apa tujuan yang ingin dicapai?

1. Data Understanding

* Informasi Dataset
* Penjelasan Tiap Fitur Pada Dataset

1. Data Preparation

* Menghapus Nilai NaN
* Mengganti nama kolom
* Melihat statistik dataset
* Menghapus kolom bernilai konstan
* Histogram Maksimum Cycle
* Membuat label untuk target prediksi
* Melihat proporsi kelas pada dataset
* Melihat korelasi antar fitur dengan heatmap
* Membagi dataset menjadi data latih dan data uji
* Proses sampling
* Proses scaling

1. Modelling & Evaluation

* Membangun model random forest (RF)
* Prediksi dan evaluasi algoritma RF
* Membangun model artificial neural network (ANN)
* Prediksi dan evaluasi algoritma ANN

Conclusion

1. **Business Understanding**

Tahap ini akan menjelaskan hal-hal yang melatar belakangi pengerjaan proyek, merumuskan masalah yang dihadapi, dan tujuan akhir yang ingin dicapai dari proyek jet engine predictive maintenance sehingga dapat menjawab masalah yang telah didefinisikan.

1. **Mengapa prediksi kerusakan mesin penting?**

Mesin jet merupakan salah satu komponen krusial yang digunakan dalam industri antariksa NASA. Mesin ini digunakan sebagai sumber tenaga dari suatu kendaraan seperti pesawat agar mampu terbang dengan adanya gaya dorong yang dihasilkan dari mesin. Melihat betapa krusialnya peran mesin pada suatu kendaraan, maka diperlukan suatu analisis yang mampu memprediksi kesehatan dari mesin apakah masih berfungsi dengan normal atau sudah mulai membutuhkan perawatan lebih lanjut. Hal ini bertujuan untuk menghindari kegagalan mesin secara mendadak yang berpotensi membahayakan kendaraan tersebut. Salah satu cara untuk mengukur performa mesin adalah dengan menggunakan sensor. Sensor-sensor ini bekerja untuk mengetahui berbagai hal seperti suhu, putaran, tekanan, vibrasi pada mesin, dan lainnya. Untuk itu, pada proyek ini akan dilakukan proses analisis untuk memprediksi kesehatan mesin berdasarkan data sensor sebelum mesin benar-benar mati.

1. **Apa masalahnya?**

Ketidaktahuan tentang kesehatan mesin dapat berpotensi menyebabkan kegagalan mesin secara mendadak saat digunakan.

1. **Apa tujuan yang ingin dicapai?**

Mengklasifikasikan kesehatan mesin ke dalam kategori normal atau failure berdasarkan data sensor.

1. **Data Understanding**

Tahap ini akan menjelaskan hal-hal yang melatar belakangi pengerjaan proyek, merumuskan masalah yang dihadapi, dan tujuan akhir yang ingin dicapai dari proyek jet engine predictive maintenance sehingga dapat menjawab masalah yang telah didefinisikan.

1. **Informasi dataset**

Dataset yang akan digunakan pada proyek ini berasal dari:

<https://data.nasa.gov/Aerospace/CMAPSS-Jet-Engine-Simulated-Data/ff5v-kuh6/about_data>

Dataset ini terdiri dari beberapa file yang secara garis besar dikelompokan menjadi 3 yaitu data train, test, dan RUL. Namun pada proyek kali ini hanya akan menggunakan data train saja yaitu train\_FD001.txt. Dataset ini memiliki 26 kolom dan 20.631 data.

1. **Penjelasan fitur**

**A black and white text on a white background

Description automatically generated**

1. **Penjelasan fitur**

Kita dapat memeriksa dimensi dan melihat data mentah sebelum diproses lebih lanjut.

[Kode untuk import library, memanggil data, dan melihat data]

A screenshot of a computer

Description automatically generated

Catatan:

* /content/train\_FD001.txt adalah lokasi dan nama dataset. Tentukan lokasi dataset pada komputer Anda.
* Data.shape mengembalikan 2 nilai yaitu jumlah data dan jumlah kolom dengan format (jumlah data, jumlah kolom).

Dari dataset kita dapat lihat bahwa nama kolom belum representatif (masih dalam bentuk angka yang berurutan) dan terdapat kolom bernilai NaN (Not a Number) pada 2 kolom terakhir. Hal ini perlu dilakukan proses lanjutan untuk membersihkan data. Proses pembersihan data akan dilakukan pada tahap selanjutnya yaitu data preparation.

1. **Data Preparation**

Tahap ini merupakan proses untuk membersihkan data sehingga output dari tahap ini yaitu data yang sudah bersih dan siap digunakan untuk proses pembuatan model *Machine Learning*. Terdapat istilah Garbage In, Garbage Out (GIGO) yang artinya jika data yang dilatih merupakan data sampah maka akan menciptakan model sampah juga. Model yang tidak bagus untuk digunakan proses prediksi. Untuk menghindari hal tersebut maka diperlukan proses data preparation. Beberapa proses yang dilakukan pada tahap ini diantaranya yaitu:

1. **Menghapus kolom bernilai NaN & Mengganti nama kolom**

Nilai NaN harus dihapus dari dataset karena nilai NaN tidak memiliki pengaruh terhadap data. Selain itu, penggantian nama kolom juga penting dilakukan agar lebih mudah dibaca dan lebih representatif.

[Kode untuk mengganti nama kolom]

A screenshot of a computer screen

Description automatically generated

Setelah dataset diberi nama kolom sesuai deskripsi, dataset terlihat lebih mudah untuk dipahami maksud dari prediktornya. Jadi, sekarang hanya terdapat 26 kolom (prediktor) dalam dataset.

1. **Melihat statistik dataset**

Proses ini dilakukan untuk mengetahui beberapa hal statistik dari data seperti nilai rata-rata, standar deviasi, nilai minimal, Q1, median, Q2, dan nilai maksimal data dari setiap kolom.

[Kode untuk melihat statistik data]

A screenshot of a computer screen

Description automatically generated

Dari data tersebut terlihat bahwa terdapat beberapa prediktor yang memiliki nilai min dan max sama besar. Hal ini menandakan bahwa prediktor tersebut memiliki nilai konstan yaitu nilai yang sama untuk semua baris. Hal ini tidak akan berpengaruh terhadap target sehingga perlu dilakukan proses lanjutan untuk menghapus prediktor tersebut untuk mengurangi beban komputasi.

1. **Menghapus kolom bernilai konstan**

Nilai konstan ditandai dengan nilai min dan max yang sama besar. Berikut fungsi untuk menghapus nilai konstan.

[Kode fungsi menghapus kolom bernilai konstan]

A screenshot of a graph

Description automatically generated

Setelah dilakukan proses penghapusan nilai konstan, dataset menyisakan 19 prediktor dari yang semula 26 prediktor. Hal ini menunjukan bahwa terdapat 7 prediktor yang memiliki nilai konstan.

1. **Membuat label untuk target prediksi**

Karena ini merupakan pekerjaan klasifikasi dan dataset belum memiliki kolom target, sehingga perlu dibuatkan kolom target secara manual. Target yang akan dibuat yaitu apakah mesin memiliki status normal atau failure (kasus klasifikasi biner). Pada proyek kali ini, status normal akan dilabeli 0 dan failure dilabeli 1.

Untuk menentukan label apakah suatu cycle sudah masuk kategori failure atau masih normal, digunakan nilai threshold 20. Artinya, untuk setiap mesin jika nilai cycle nya sudah mencapai (maksimal cycle - threshold), maka cycle tersebut akan dilabeli sebagai failure. Misalnya, engine 1 memilili maksimal cycle 120. Maka cycle 101 s.d. 120 akan dilabeli sebagai failure. Hal ini dilakukan untuk mencegah mesin benar-benar mati sehingga dapat dilakukan perawatan dan persiapan penggantian mesin lebih dini.

Berikut fungsi untuk membuat label status mesin.

[Kode fungsi assign label]

A group of people with different colored lines

Description automatically generated with medium confidence

1. **Melihat korelasi fitur dengan heatmap**

Nilai pengaruh atau dikenal sebagai nilai korelasi dalam dataset dapat dibagi menjadi 5 kategori yaitu:

A table of numbers with black text

Description automatically generated

Untuk melihat nilai korelasi antara prediktor dengan target, maka digunakan visualisasi heatmap. Pada proyek ini, akan digunakan nilai threshold 0.20.

A colorful grid with numbers

Description automatically generated

Dari visualisasi heatmap di atas, hanya prediktor yang memiliki nilai absolut korelasi lebih dari atau sama dengan threshold yang akan ditampilkan. Nilai theshold 0.2 digunakan karena nilai korelasi yang di atas angka 0.2 merupakan korelasi yang cukup kuat. Sedangkan di bawah 0.2 masuk kategori sangat lemah sehingga tidak perlu digunakan.

Nilai negatif pada korelasi menunjukan bahwa prediktor tersebut memiliki korelasi yang berkebalikan dengan prediktor lainnya. Misalnya sensor 2 dan sensor 7 yang memiliki nilai korelasi -0.7. Artinya, ketika nilai pada sensor 2 meningkat maka nilai sensor 7 akan menurun dan sebaliknya. Semakin tinggi nilai korelasi, maka saling mempengaruhi satu sama lain. Nilai absolut nilai korelasi yaitu antara 0 s.d. 1. Nilai 0 artinya tidak korelasi sedangkan 1 artinya korelasi sangat kuat.

1. **Seleksi fitur**

Dalam beberapa kasus, tidak semua prediktor (kolom) pada dataset memilki pengaruh yang cukup kuat terhadap target. Untuk itu, perlu dilakukan proses seleksi fitur untuk membuang fitur yang tidak memiliki pengaruh. Tujuannya adalah untuk mengurangi waktu dan beban komputasi yang digunakan dalam proses pembelajaran.

Sesuai pada tahap sebelumnya, akan digunakan nilai threshold 0.2. Sehingga prediktor yang memiliki nilai korelasi < 0.2 akan dihapus. Berikut fungsi untuk seleksi fitur.

[Kode untuk seleksi fitur]

A table with numbers and symbols

Description automatically generated

Setelah proses seleksi fitur, tersisa 15 kolom yang terdiri dari 14 prediktor dan 1 target.

1. **Melihat proporsi kelas pada dataset**

Langkah selanjutnya yaitu melihat proporsi kelas pada dataset. Kita akan melihat proporsi kelas normal (0) dan failure (1). Hal ini dilakukan untuk mengetahui keseimbangan dataset.

[Kode untuk melihat keseimbangan dataset]

A graph with a bar

Description automatically generated

Dari visualisasi di atas dapat diketahui bahwa data cycle mesin yang tergolong normal sejumlah 18.631 cycle dan failure sejumlah 2.000 cycle. Dari jumlah tersebut dapat dikatakan bahwa proporsi nilai minoritas yaitu 9.7% dari total data dalam dataset. Proporsi tersebut masuk pada kategori moderate sehingga perlu dilakukan proses sampling untuk menambah data pada data minoritas. Fenomena ini disebut sebagai dataset tidak seimbang. Artikel mengenai dataset tidak seimbang dapat dilihat pada laman:

<https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data>

1. **Membagi dataset menjadi data latih dan data uji**

Sebelum proses penyeimbangan data dilakukan (proses sampling), data perlu dibagi dahulu menjadi 2 bagian yaitu data train dan data test. Data train akan digunakan dalam pembuatan model machine learning sedangkan data test digunakan untuk melihat performa model machine learning yang dihasilkan.

Pada proyek ini, proses pembagian dataset akan menggunakan skema 80:20 yang artinya 80% data akan digunakan sebagai data train dan 20% data digunakan sebagai data test. Tidak ada aturan khusus untuk skema yang digunakan. Beberapa proyek menggunakan skema 60:40, 70:30, 75:25, 80:20, dan 90:10. Namun 1 hal yang pasti adalah jumlah data test tidak boleh melebihi data train. Selain itu, akan dibagi juga antara kolom prediktor (disimbolkan dengan awalan X) dan kolom target (disimbolkan dengan awalan y).

A diagram of data training

Description automatically generated

[Kode untuk melihat split dataset]

Setelah dataset dibagi, kita lihat jumlah data train dan data test dengan menggunakan fungsi shape.

[Kode untuk melihat dimensi data train dan test]

Dari total 20.631 data pada dataset, dapat diketahui bahwa jumlah data untuk proses training sebanyak 16.504 data dan jumlah data untuk proses testing sebanyak 4.127 data. Angka 14 menunjukan bahwa terdapat 14 prediktor yang akan dicari polanya selama proses pembelajaran.

1. **Proses sampling**

Proses sampling digunakan untuk mengatasi masalah dataset tidak seimbang. Tujuan dari proses ini adalah untuk menyeimbangkan proporsi kelas pada dataset sehingga kelas normal dan failure akan memiliki jumlah data yang sama. Hal ini akan membuat model machine learning sensitif terhadap kedua kelas data (normal dan failure) tidak hanya pada salah satunya saja.

Proses sampling hanya boleh dilakukan pada data train untuk mencegah kebocoran data dari data test. Oleh karena itu, pada tahap sebelumnya dilakukan proses pembagian data train dan data test terlebih dahulu.

Teknik sampling yang akan digunakan pada proyek ini yaitu oversampling di mana data minoritas (failure) akan dibuatkan data sintetis sehingga jumlahnya akan sama dengan kelas mayoritas (normal). Untuk algoritma yang digunakan yaitu Synthetic Minority Oversampling Technique (SMOTE). Baca lebih lanjut mengenai SMOTE pada link berikut:

<https://medium.com/@corymaklin/synthetic-minority-over-sampling-technique-smote-7d419696b88c>

[Kode untuk proses sampling]

A graph with blue squares

Description automatically generated

Terlihat pada barplot di atas dapat diketahui bahwa setelah proses oversampling, jumlah data antara mesin normal dan failure sudah seimbang dengan masing-masing status memiliki 14.861 data.

1. **Proses scaling**

Sama seperti proses sampling, proses scaling juga hanya boleh dilakukan pada data train untuk mencegah kebocoran data dari data test. Selain itu, proses scaling juga dilakukan setelah proses sampling. Tidak boleh terbalik. Oleh karena itu, pada tahap sebelumnya dilakukan proses pembagian data train dan data test terlebih dahulu, kemudian dilanjutkan dengan proses sampling, dan terakhir adalah scaling.

Proses scaling digunakan untuk menyamakan rentang nilai dari semua fitur. Hal ini bertujuan untuk mengurangi beban komputasi saat proses training dan meningkatkan performa model yang dihasilkan. Proses scaling dilakukan jika ada prediktor yang memiliki nilai jauh di atas nilai prediktor lainnya. Misalnya nilai pada sensor 11 bernilai puluhan sedangkan sensor 21 bernilai puluhan ribu.

Pada proyek ini akan digunakan metode Z-Score untuk proses scaling. Informasi lebih lanjut mengenai normalisasi Z-Score dapat dilihat pada link berikut:

<https://www.statology.org/z-score-normalization/>

[Kode untuk proses scaling]

A screenshot of a computer code

Description automatically generated

Dari hasil scaling terlihat bahwa untuk semua prediktor memiliki jangkauan data yang tidak berbeda jauh. Hal ini akan memudahkan dalam proses pembuatan model machine learning dan mengurangi waktu dan resource komputasi yang dibutuhkan.

1. **Modelling & Evaluation**

Tahap ini merupakan proses untuk membuat model machine learning yang nantinya akan digunakan untuk proses prediksi. Beberapa hal yang dilakukan pada tahap ini yaitu:

* Memilih algoritma machine learning yang akan digunakan dan tuning hyperparameter.
* Proses fitting atau proses model belajar.
* Proses evaluasi model untuk mengetahui performa model.

Output dari tahap ini adalah model yang telah dilatih dan siap digunakan untuk proses prediksi.

1. Membuat model dari algoritma random forest (RF)

Random forest merupakan algoritma klasifikasi yang cukup populer karena performanya yang sangat baik. Pada artikel ini tidak membahas detail dari random forest sehingga Anda dapat membaca lebih lanjut mengenai random forest pada sumber berikut:

<https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/>

Setelah data dibersihkan pada proses pre-processing, selanjutnya yaitu membangun model machine learning. Untuk membuat model ML dari random forest, akan digunakan library yang sudah disediakan oleh scikit-learn.

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

[Kode untuk modelling dengan random forest]

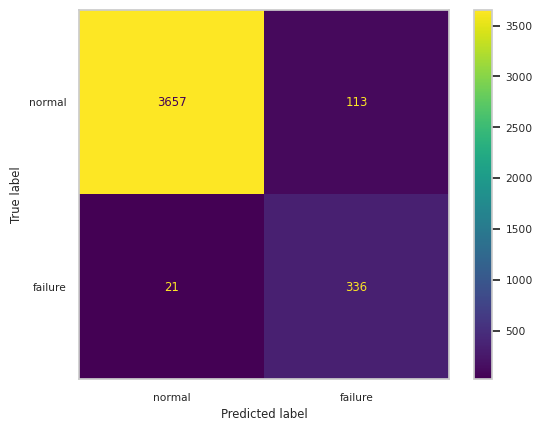
Catatan:

* Fungsi RandomForestClassifier() merupakan fungsi dari libary scikit-learn yang digunakan untuk membuat model ML dengan algoritma random forest.
* Fungsi fit() digunakan untuk proses pelatihan dan pembelajaran mesin sehingga akan tercipta model ML. Fungsi fit() membutuhkan 2 data yaitu X\_train dan y\_train. X\_train merupakan data yang berisi data prediktor sedangkan y\_train berisi data target.
* Fungsi dump() dari joblib ini digunakan untuk menyimpan model random forest. Model hasil pelatihan perlu kita export untuk ditanamkan pada suatu aplikasi untuk memprediksi data baru.
* Fungsi predict() digunakan untuk memprediksi data baru. Fungsi ini memerlukan satu data yaitu X\_test yang merupakan data prediktor untuk data testing. Hasil fungsi ini adalah prediksi target dari X\_test yang selanjutnya disimpan pada variabel y\_predict.

Setelah berhasil memprediksi data menggunakan fungsi predict(), kemudian kita akan evaluasi hasil prediksi untuk mengetahui apakah model yang dihasilkan sudah bagus atau belum. Untuk melakukan evaluasi, ada beberapa ukuran evaluasi yang akan digunakan yaitu akurasi, precision, recall, dan f1 score. Sebelum menghitung nilai evaluasi, terlebih dahulu akan digunakan confusion matrix untuk mengetahui nilai True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN). Informasi lebih lanjut mengenai confusion matrix dapat dilihat pada artikel berikut:

<https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>

[Kode untuk confusion matrix].



Dari tabel confusion matrix di atas, dapat diketahui hal-hal sebagai berikut:

* True Positive (TP) -> Cycle failure yang benar diprediksi failure. Ada sebanyak 336 data.
* True Negative (TN) -> Cycle normal yang benar diprediksi normal. Ada sebanyak 3.657 data.
* False Positive (FP) -> Cycle normal yang diprediksi failure. Ada sebanyak 113 data.
* False Negative (FN) -> Cycle failure yang diprediksi normal. Ada sebanyak 21 data.

[Kode untuk menghitung skor evaluasi]

A screenshot of a computer code

Description automatically generated

Dari nilai evaluasi di atas, dapat disimpulkan sebagai berikut:

* Nilai accuracy menunjukan bahwa model mampu memprediksi 96% data dengan benar. Dengan kata lain, dari 4.127 data uji model dapat dengan benar memprediksi 3.989 data.
* Nilai precision menunjukan bahwa dari semua cycle yang diprediksi failure oleh model, ternyata hanya 74% yang benar. Dengan kata lain, dari 449 cycle yang diprediksi failure ternyata hanya 336 cycle yang benar-benar berstatus failure. Sisanya berstatus normal.
* Nilai recall menunjukan bahwa dari semua cycle yang aktualnya memang benar-benar berstatus failure, ternyata hanya 94% yang berhasil diprediksi failure oleh model. Dengan kata lain, dari 357 cycle yang memang failure, mampu memprediksi dengan benar sebanyak 337 cycle. Hanya 20 cycle berstatus failure yang terprediksi normal oleh model.
* Nilai F1 menunjukan bahwa model mampu mengenali kondisi cycle mesin masih normal dan failure dengan baik. Tidak condong ke salah satu kondisi saja.

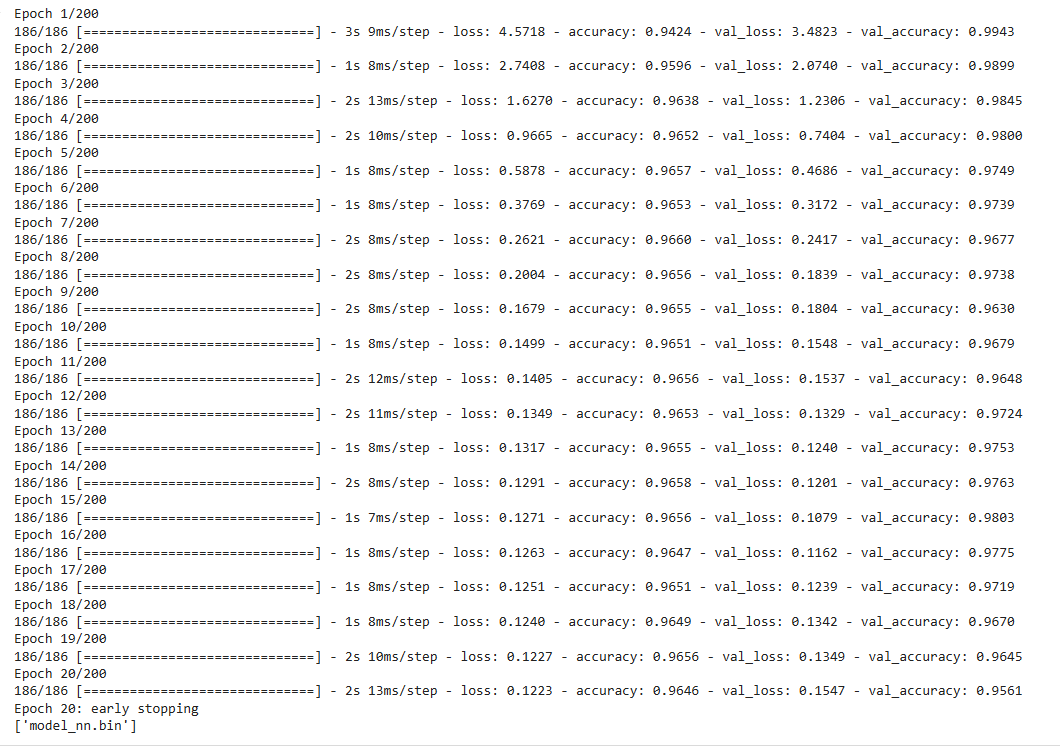
1. Membuat model dari algoritma artificial neural network (ANN)

ANN merupakan salah satu algoritma machine learning yang menjadi cikal bakal algoritma deep learning. Disebut neural karena cara kerja dari ANN yaitu meniru bagaimana cara kerja neuron pada otak manusia untuk mentransfer sinyal ke neuron yang lain. Pembahasan lebih lanjut mengenai ANN dapat dilihat pada artikel:

<https://www.analyticsvidhya.com/blog/2021/09/introduction-to-artificial-neural-networks/>

Pada proyek ini akan digunakan library dari Tensorflow untuk membuat model ANN. Berikut adalah kode untuk membangun arsitektur ANN.

[Kode untuk ANN]



Algoritma Neural Network yang digunakan memiliki arsitektur sebagai berikut:

* Jumlah layer => 5 yang terdiri dari 1 input layer, 3 hidden layer, dan 1 output layer.
* Input layer memiliki 14 neuron. Angka ini disesuaikan dengan banyaknya prediktor pada data train.
* Hidden layer 1, 2, dan 3 masing-masing memiliki 512, 256, dan 128 neuron.
* Output layer memiliki 1 neuron dengan fungsi aktivasi sigmoid. Hal ini memungkinkan untuk menghasilkan output berupa nilai pecahan antara 0 s.d. 1. Pada project ini menggunakan threshold 0.5. Jika nilai output >= 0.5 maka failure dan jika < 0.5 maka normal.
* Arsitektur ini menggunakan fungsi optimizer ADAM. Fungsi ini digunakan untuk mengatur bobot tiap neuron pada proses pembelajaran.
* Fungsi loss yang digunakan yaitu binary\_crossentropy. Fungsi ini digunakan untuk menghitung nilai error pada output layer. Nilai error didapat dari selisih antara data aktual dan data prediksi.
* Metrik evaluasi yang diukur selama proses pembelajaran mesin yaitu nilai akurasi.
* Proses pembelajaran ini menggunakan fungsi EarlyStopping() yang berfungsi untuk menghentikan proses pembelajaran jika model tidak bertambah baik untuk waktu tertentu.

**Catatan:**

Secara umum, proses pembelajaran (training) pada neural network terdiri dari 2 proses utama yaitu forward dan backward.

* Forward => Proses untuk menghasilkan nilai output. Proses ini dimulai dari input layer > hidden layer > output layer. Di layer output, kemudian dilakukan proses perhitungan nilai loss untuk mengetahui nilai error. Seberapa jauh nilai prediksi dengan nilai aktualnya. Jika nilai error lebih besar dari ambang batas yang ditentukan, maka dilakukan proses backward.
* Backward => Proses untuk mengupdate bobot. Jika nilai error yang dihasilkan terlalu besar, maka dilakukan proses backward untuk mengupdate bobot pada tiap neuron pada tiap layer. Proses ini dilakukan dalam rangka menurunkan nilai error yang dihasilkan.

Penjelasan dari beberapa istilah:

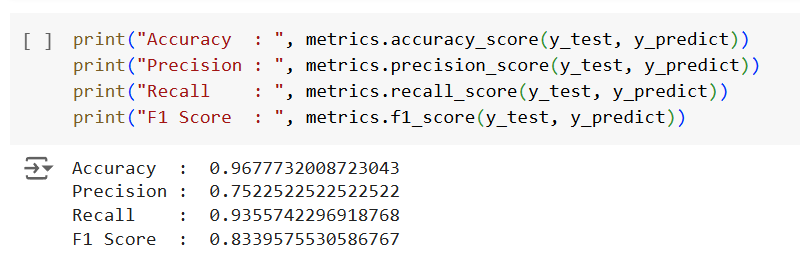
* Learning rate => Seberapa besar bobot pada neuron berubah selama proses training dalam rangka meminimalkan nilai error. Nilai LR memiliki range antara 0 s.d. 1. Semakin besar nilai LR, semakin cepat proses training. Begitupun sebaliknya semakin kecil nilai LR, maka proses training semakin lambat. Diperlukantuning hyperparameteruntuk mencari nilai LR yang tepat.
* Epochs => Hyperparameter yang menentukan berapa kali algoritma deep learning bekerja melewati seluruh dataset baik secara forward (maju) maupun backward (mundur). Forward berfungsi untuk menentukan nilai prediksi/output. Sedangkan backward berfungsi untuk mengupdate bobot jika nilai error yang dihasilkan dari proses forward terlalu besar (melebihi threshold).
* Batch size => Merupakan jumlah sampel data yang disebarkan ke Neural Network. Misalnya ada 100 data dan memiliki batch size 10. Maka 10 data pertama akan disebarkan ke neural network untuk proses pembelajaran. Kemudian 10 data berikutnya. Begitu seterusnya sampai 100 data tersebar semua. Setiap 1x data tersebar semua, maka dianggap 1 epochs.
* Optimizer => Fungsi yang digunakan untuk melakukan update bobot pada proses backward. Nilai bobot ini perlu diupdate dalam rangka meminimalkan nilai error.
* Loss => Fungsi yang digunakan untuk menghitung nilai error. Jika nilai error > threshold, maka akan dilakukan proses backward untuk mengupdate bobot. Proses forward dan backward ini dilakukan secara terus menerus sampai tercapai 2 kondisi yaitu tercapainya maksimal epochs atau nilai error < threshold yang ditentukan.

Setelah proses training selesai, sama seperti pada Random Forest, akan dilakukan proses evaluasi untuk melihat performa model ANN. Berikut kode confusion matrix dari ANN.

[Kode confusion matrix untuk ANN]

A screenshot of a chart

Description automatically generated



Dari nilai evaluasi di atas, dapat disimpulkan sebagai berikut:

* Nilai accuracy menunjukan bahwa model mampu memprediksi 96% data dengan benar. Dengan kata lain, dari 4.127 data uji model dapat dengan benar memprediksi 3.992 data.
* Nilai precision menunjukan bahwa dari semua cycle yang diprediksi failure oleh model, ternyata hanya 75% yang benar. Dengan kata lain, dari 449 cycle yang diprediksi failure ternyata hanya 338 cycle yang benar-benar berstatus failure. Sisanya berstatus normal.
* Nilai recall menunjukan bahwa dari semua cycle yang aktualnya memang benar-benar berstatus failure, ternyata hanya 93% yang berhasil diprediksi failure oleh model. Dengan kata lain, dari 357 cycle yang memang failure, mampu memprediksi dengan benar sebanyak 335 cycle. Hanya 22 cycle berstatus failure yang terprediksi normal oleh model.
* Nilai F1 menunjukan bahwa model mampu mengenali kondisi cycle mesin masih normal dan failure dengan baik. Tidak condong ke salah satu kondisi saja.

**Conclusion**

Secara garis besar, artikel ini membahas tentang klasifikasi mesin jet apakah masuk kategori normal atau failure berdasarkan data sensor. Beberapa tahapan yang perlu dilakukan dalam membuat model ML yaitu dari tahap business understanding untuk melihat masalah yang akan dipecahkan, data understanding untuk memahami data, data preparation untuk menyiapkan dan membersihkan data sehingga menjadi data yang baik dan siap untuk proses modelling, modelling untuk membuat model ML, dan evaluation untuk mengetahui apakah model yang dihasilkan sudah baik atau belum. Dari percobaan yang telah dilakukan, dapat ditarik kesimpulan bahwa model yang dihasilkan oleh Random Forest dan Artificial Neural Network sudah bagus karena memiliki nilai evaluasi yang tinggi sehingga dapat digunakan untuk proses prediksi pada data baru. Anda dapat melakukan tahap terakhir dari CRISP-DM yaitu deployment yang tidak dibahas di sini. Anda dapat menggunakan model yang telah di export untuk proses prediksi data baru yang ditanamkan pada suatu aplikasi. Good luck!