Machine Learning Evaluation & Supervised Learning

Getting Started :

- Split Data Train & Test
  - Modelling
  - Model Evaluation: metrics model
  - Model Evaluation: Apakah model sudah best-fit? Hindari Overfit/Underfit. Validasi dengan cross-validation
  - Hyperparameter Tuning
  - Feature Important

A. How to Split Train & test

Train/test split adalah salah satu metode yang dapat digunakan untuk mengevaluasi performa model machine learning. Metode evaluasi model ini membagi dataset menjadi dua bagian yakni bagian yang digunakan untuk training data dan untuk testing data dengan proporsi tertentu

```
from sklearn.model_selection import train_test_split
x = df.drop(columns=['Reached.on.Time_Y.N'], axis=1)
y = df['Reached.on.Time_Y.N']
```

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

```
Preprocessing data train lanjutan
```sh
y_train.value_counts(normalize=True)
```

```
# oversampling
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state = 2)
x_over, y_over = sm.fit_resample(x_train, y_train.ravel())
```

B.Modelling

Memilih model untuk machine learning dengan melihat dari masing-masing classifire mana yang terbaik,dari sini kami memilih XGboost dengan nilai = ROC_AUC 71,85%

```
from sklearn.metrics import accuracy_score, recall_score, roc_auc_score,
precision_score, f1_score
from sklearn.ensemble import StackingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
```
```

```
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from sklearn.ensemble import AdaBoostClassifier
def algorithm_pipeline(X_train_data, X_test_data, y_train_data, y_test_data, model):
  best_model = model
  model.fit(X_train_data, y_train_data)
  pred = model.predict(X_test_data)
  score = roc_auc_score(y_test_data, pred)
  return [best_model, pred, score]
models_to_train = [LogisticRegression(),
                   KNeighborsClassifier(),
                   DecisionTreeClassifier(),
                   RandomForestClassifier(),
                   AdaBoostClassifier(),
                   XGBClassifier(),
                   LGBMClassifier()
                  ]
models_preds_scores = []
for i, model in enumerate(models_to_train):
    result = algorithm_pipeline(x_over, x_test, y_over, y_test,  model)
    models_preds_scores.append(result)
for result in models_preds_scores:
    print('Model: {0}, Score: {1}'.format(type(result[0]).__name__, result[2])) #
score in training data
```

Model Evaluation : Pemilihan dan perhitungan metrics model

dari sini kita melakukan model evaluasi dan melakukan perhitungan model.

```
def eval_classification(model):
    y_pred = model.predict(x_test)
    y_pred_train = model.predict(x_over)
    print("Accuracy (Test Set): %.2f" % accuracy_score(y_test, y_pred))
    print("Precision (Test Set): %.2f" % precision_score(y_test, y_pred))
    print("Recall (Test Set): %.2f" % recall_score(y_test, y_pred))
    print("F1-Score (Test Set): %.2f" % f1_score(y_test, y_pred))
    print("roc_auc (test-proba): %.2f" % roc_auc_score(y_test, y_pred))
    print("roc_auc (train-proba): %.2f" % roc_auc_score(y_over, y_pred_train))
def confusionmatrix(predictions):
    cm = confusion_matrix(y_test, predictions)
    disp = ConfusionMatrixDisplay(confusion_matrix=cm)
    return disp.plot()
 eval_classification(XGB)
```

Confusion Matrix

Confusion Matrix adalah pengukuran performa untuk masalah klasifikasi machine learning dimana keluaran dapat berupa dua kelas atau lebih

```
import xgboost as xgb
from sklearn.metrics import classification_report, confusion_matrix,
ConfusionMatrixDisplay
predictions = XGB.predict(x_test)
confusionmatrix(predictions)
```

D.Model Evaluation: Apakah model sudah best-fit? Hindari Overfit/Underfit. Validasi dengan cross-validation

Cross-validation (CV) adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi

```
cross_validation = XGBClassifier(cv=5)
cross_validation.fit(x_over, y_over)
eval_classification(cross_validation)
```

E. Hyper Tunning

Model hyperparameter adalah pengaturan eksternal yang nilainya tidak dapat ditebak dari data.

```
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
import numpy as np
#Menjadikan ke dalam bentuk dictionary
hyperparameters = {
                    'max_depth' : [int(x) for x in np.linspace(10, 110, num = 11)],
                    'min_child_weight' : [int(x) for x in np.linspace(1, 20, num =
11)],
                    'gamma' : [float(x) for x in np.linspace(0, 1, num = 11)],
                    'tree_method' : ['auto', 'exact', 'approx', 'hist'],

                    'colsample_bytree' : [float(x) for x in np.linspace(0, 1, num =
11)],
                    'eta' : [float(x) for x in np.linspace(0, 1, num = 100)],

                    'lambda' : [float(x) for x in np.linspace(0, 1, num = 11)],
                    'alpha' : [float(x) for x in np.linspace(0, 1, num = 11)]
                    }
# Init
from xgboost import XGBClassifier
xg = XGBClassifier(random_state=42)
xg_tuned = RandomizedSearchCV(xg, hyperparameters, cv=5, random_state=42,
scoring='precision')
xg_tuned.fit(x_over,y_over)
# Predict & Evaluation
eval_classification(xg_tuned)
```

2. Feature Importance

Feature Selection adalah suatu kegiatan pemodelan atau penganalisaan data yang umumnya dapat dilakukan secara preprocessing dan bertujuan untuk memilih fitur yang berpengaruh (fitur optimal) dan mengesampingkan fitur yang tidak berpengaruh

```
feature_important = XGB.get_booster().get_score(importance_type='weight')
keys = list(feature_important.keys())
values = list(feature_important.values())
data = pd.DataFrame(data=values, index=keys, columns=["score"]).sort_values(by =
"score", ascending=True)
data.nlargest(40, columns="score").plot(kind='barh', figsize = (20,10)) ## plot top 40
features
```

```
Pada modeling ini kita memilih model terbaik yaitu XGBClassifier, model ini memiliki
nilai precision 88% setelah kita hyperparameter tuning.
Fitur penting yang digunakan dalam modelling:
- hampir semuanya feature digunakan pada modeling kecuali ID
Business Insight
1. Cukup banyak keterlambatan yang terjadi pada pesanan barang dengan diskon diatas
10%.
2. Berat barang pada rentang 2-4 kg mengalami keterlambatan yang terlampau sering.
3. Terjadi penumpukan barang di Warehous F, sehingga terjadi banyaknya barang yang
telat dari warehouse F
4. Jumlah Customer care calls mempengaruhi apakah barang akan terlambat atau tidak,
dimana berdasarkan pengerjaan tugas yang lalu semakin banyak customer care calls yang
dilakukan pelanggan maka potensi barang datang terlambat cukup besar.
Solusi:
1. Notifikasi otomatis melalui pesan singkat atau media telekomunikasi lainnya apabila
pengiriman diprediksi akan terlambat.
2. Sebagai ganti rugi atas keterlambatan pengiriman, perusahaan dapat memberikan kupon
gratis. Misalnya, kupon gratis ongkir hingga X nominal.
3. Memberikan Rekomendasi kepada customer Mode of Shipment berdasarkan berat dan
product importance.
4. Mengkoordinir setiap warehouse supaya tidak terjadi penumpukan barang di setiap
warehousenya, sehingga barang keluar warehouse tidak banyak birokrasi yang berbelit-
belit.
5. Sebelum mengirimkan barang, maka alangkah baiknya diuji terlebih dahulu apakah
barang akan terlambat atau tidak, jika terlambat maka estimasi waktu pengiriman dapat
ditambah.
```

Source

Source	Link
Kaggle Dataset Download	[https://www.kaggle.com/datasets/prachi13/customer-analytics]
Google Collabs	[https://colab.research.google.com/drive/1tY9_lL3l0owtlQnWhSdcabRIhjyDkN6usp=sharing#scrollTo=9_-yhwZhyFx0]