Buchanan Venture Partners

Improving Employee Retention by Predicting Employee Attrition Using Machine Learning

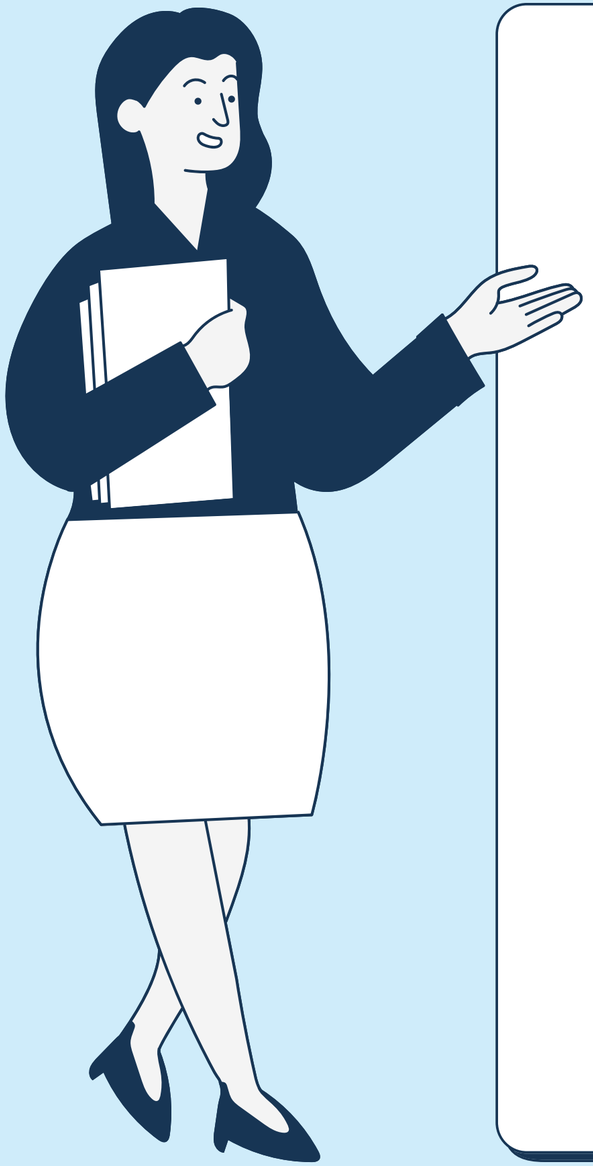
Sponsorship opportunities

Fadillah Akbar

Data Scientist

Data scientist with a strong mathematical foundation and problem-solving abilities. Experienced in building projects in data mining, data processing, business performance analysis, data visualization, and predictive modeling across multiple industries. Motivated to use data science to improve business impact through analytics, statistics, and machine learning.

www.linkedin.com/in/fadillahakbar



Business Overview

Human resources (HR) is the main asset that needs to be managed properly by the company so that business goals can be achieved effectively and efficiently. On this occasion, we will face a problem about human resources in the company. Our focus is to find out how to keep employees to stay in the current company which can result in increased costs for employee recruitment and training for those who have just entered. By knowing the main factors that cause employee dissatisfaction, the company can immediately address them by creating programs that are relevant to employee problems.

Table of Content

What we'll discuss

www.linkedin.com/in/fadillahakbar

Data Preprocessing

Making raw data into clean and ready-to-process data

www.linkedin.com/in/fadillahakbar

Data Preprocessing

Null Data

There are 6 columns with missing values, one of which will be removed (>50% missing data), and the others will be imputed using the data measurement scale.

```
import statistics

# Drop IkutProgramLOP because missing value more than 50%
df = df.drop(["IkutProgramLOP"],axis=1)

# SkorKepuasanPegawai is a variable with Ordinal measurement scale, so mode is the best imputation.
df["SkorKepuasanPegawai"] = df["SkorKepuasanPegawai"].fillna(statistics.mode(df["SkorKepuasanPegawai"]))

# AlasanResign is filled with 'still_working' because those who do not have a resignation reason, the employee is still there.
df["AlasanResign"] = df["AlasanResign"].fillna("masih_bekerja")

# The other variables are ratio scaled, so the median is the best imputation.
df["JumlahKeikutsertaanProjek"] = df["JumlahKeikutsertaanProjek"].fillna(df["JumlahKeikutsertaanProjek"].median())
df["JumlahKeterlambatanSebulanTerakhir"] = df["JumlahKeterlambatanSebulanTerakhir"].fillna(\
                                           df["JumlahKeterlambatanSebulanTerakhir"].median())
df["JumlahKetidakhadiran"] = df["JumlahKetidakhadiran"].fillna(df["JumlahKetidakhadiran"].median())
```

Data Preprocessing

Incorrect Value

There is data that is wrongly inputted, it should be a number but is inputted with a word

```
# Change the incorrect value in the PernahBekerja
df["PernahBekerja"] = df["PernahBekerja"].replace("yes",1)
df["PernahBekerja"].value_counts()
```

Unnecessary Data

It turns out that this variable contains all the same (constant variable), so it needs to be discarded

```
# Membuang variabel PernahBekerja
df = df.drop(["PernahBekerja"],axis=1)
```

www.linkedin.com/in/fadillahakbar

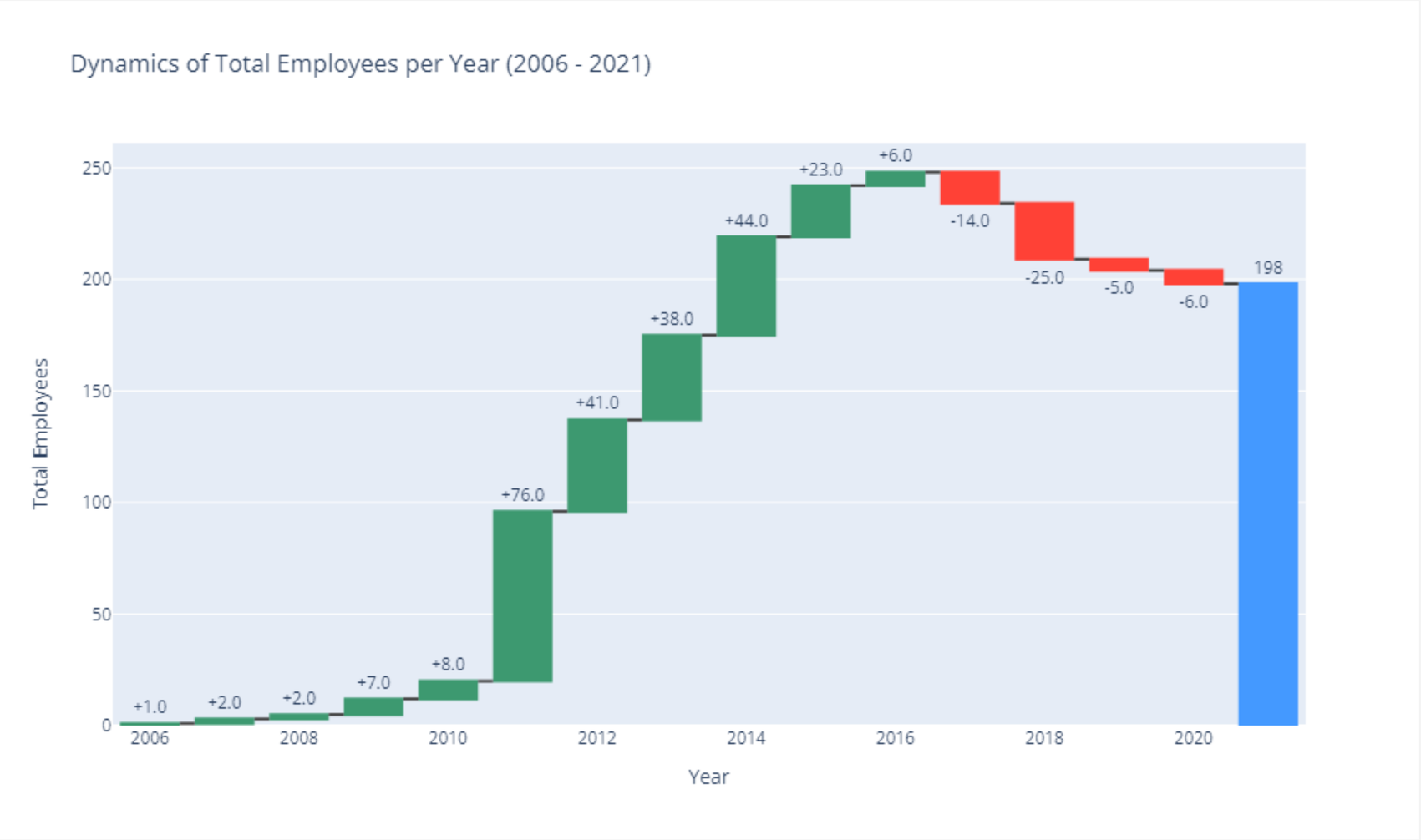
Annual Report on Employee Number Changes

Analyze the development of the number of employees in the company every year so as to get the current condition of the company whether it is in a stable state.

www.linkedin.com/in/fadillahakbar

www.linkedin.com/in/fadillahakbar

Annual Report on Employee Number Changes



Insight/Analysis

This analysis focuses on the trend of changes in the number of employees from year to year.

Since the company first began employing in 2006, the number of employees has climbed by 250% per year for the next decade, with the largest increase in 2011.

Since 2017, the number of employees who departed has outnumbered the number of employees who joined, and this trend has continued until 2020, when only 198 people remained, resulting in a 20% fall in total employees.

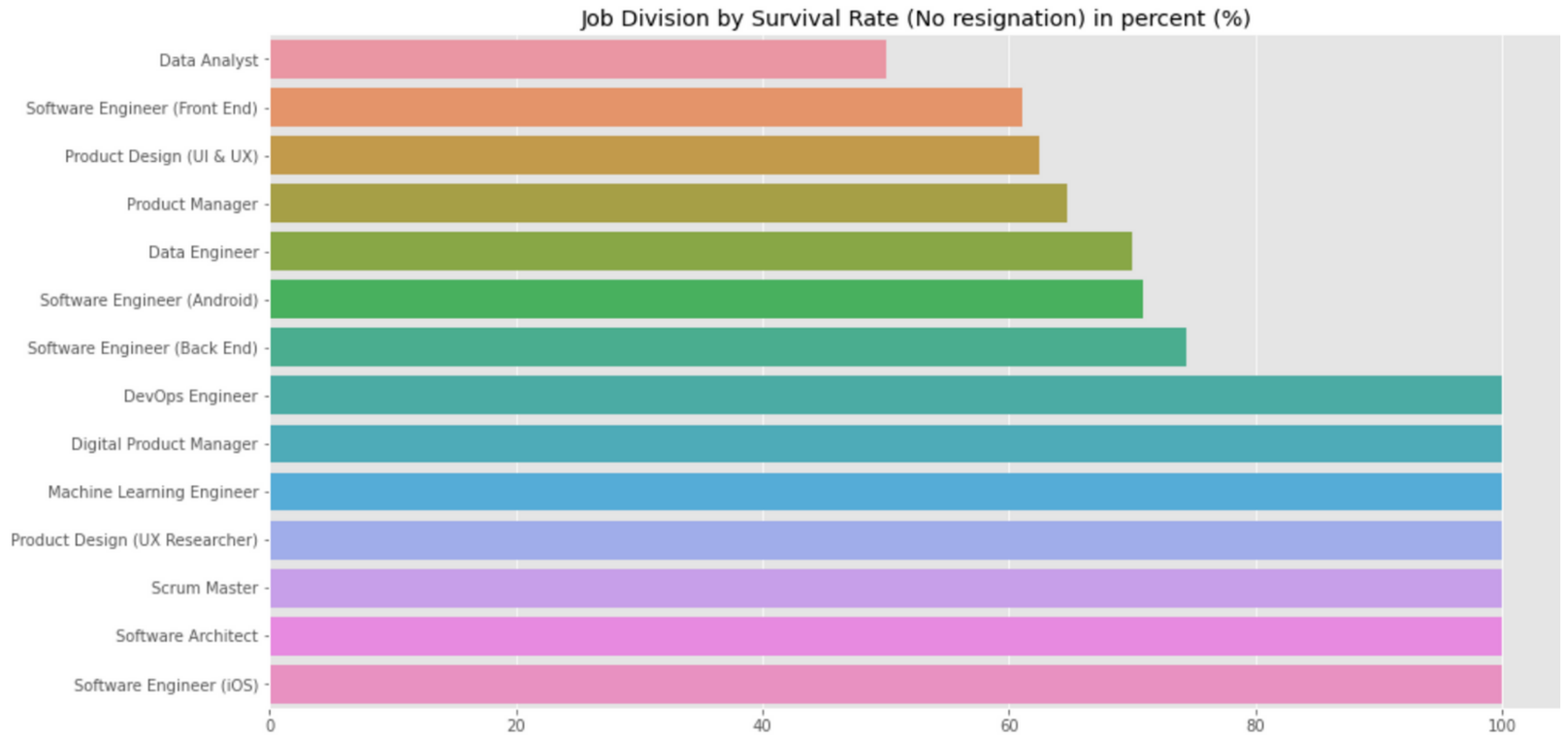
Resign Reason Analysis for Employee Attrition Management Strategy

Analyze what job divisions are most prone to resign and what reasons for resignation can be handled by the company.

www.linkedin.com/in/fadillahakbar

www.linkedin.com/in/fadillahakbar

Job Divisions with the Highest Resignation Rate

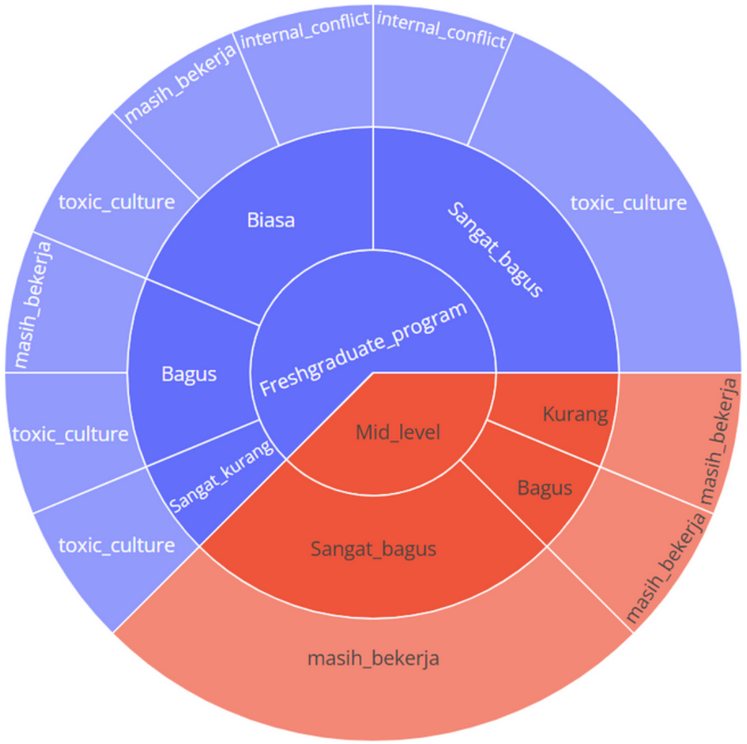


Insight/Analysis

This investigation focuses on the working conditions of employees in the division with the highest turnover rate.

When comparing all divisions, the Data Analyst Division has the highest resignation rate (50%).

Data Analyst Division by Career Ladder, Performance, and Recent Company Conditions



Insight/Analysis

During the investigation, it was discovered that all of the employees who quit were Fresh Graduate Program employees who, regardless of the findings of the performance evaluation, must have been the cause of the "toxic culture" in this division. So that it can be determined that there is mismanagement by the division's leadership level, the company should change the team or division supervisor so that there are no more new employees who resign because of the division's culture.

www.linkedin.com/in/fadillahakbar

Build an Automated Resignation Behavior Prediction using Machine Learning

Create the best machine learning model by following the rules of human-centered AI (responsible for the features used) to find out employee problems early, so that companies can map employees who resign in the retention program as early as possible.

www.linkedin.com/in/fadillahakbar

www.linkedin.com/in/fadillahakbar

Machine Learning Preparation

1. Data Preprocessing

2. Model Training

3. Model Selection

4. Hyperparameter Tuning

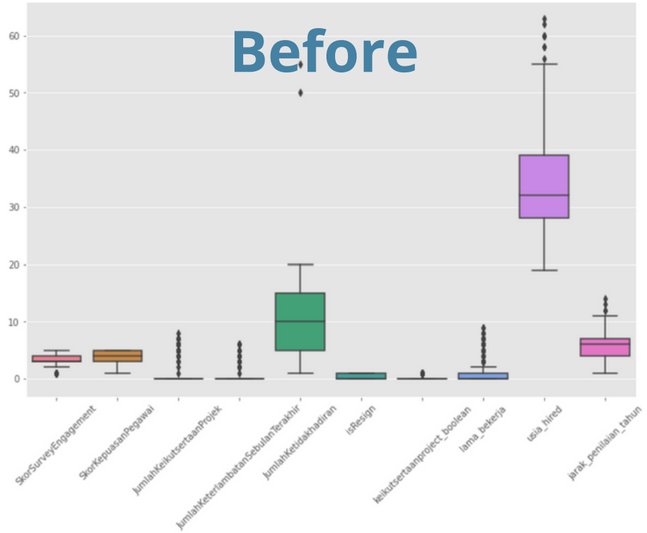
Data Preprocessing

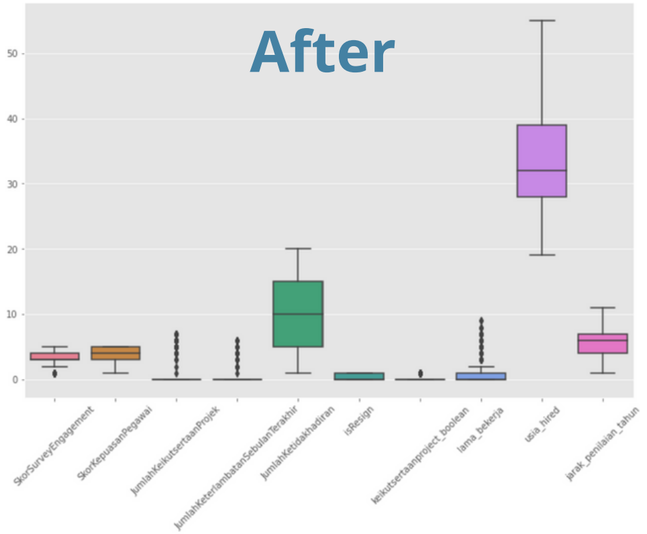
```
# Missing Value Detection
missing_value_columns = []

# Iteration
for i in df_ml.columns:
    # Take the unique value in each column that is iterated
    column_loop = df_ml[i].unique().tolist()
    # Two-way check
    # 1. Is there a " " in the column
    # 2. Is there nan in the column
    if " " in column_loop or sum([1 if str(x) == "nan" else 0 for x in column_loop]) > 0:
        # If there is one, append the column name here
        missing_value_columns.append(i)

# Print the content
print(missing_value_columns)
```

```
[]
```

```
# Check whether the rows are duplicated or not
df_ml.duplicated().sum()
```

```
0
```





At this point, a search is conducted to determine whether the data contains missing values, duplicates, or outliers. There were no missing values or duplicates in the data, but there were outliers in certain observations. As a result, we used the IQR approach to remove outlier data. Outliers are accepted in some variables due to their statistical significance, such as low standard deviation or dichotomous data (two unique values, such as yes and no).

www.linkedin.com/in/fadillahakbar

Data Preprocessing

```python
# Data Repairing and Domain-Expertise Encoding
df_ml = df_clean.copy()

df_ml["StatusPernikahan"] = df_ml["StatusPernikahan"].map(lambda x: "Lainnya" if x == "-" else x)
df_ml["AsalDaerah"] = df_ml["AsalDaerah"].map(lambda x: x.replace(" ", "_"))

education_class = {"Sarjana": 1,
                   "Magister": 2,
                   "Doktor": 3}
df_ml["TingkatPendidikan"] = df_ml["TingkatPendidikan"].map(education_class)

career_level = {"Freshgraduate_program": 1,
                "Mid_level": 2,
                "Senior_level": 3}
df_ml["JenjangKarir"] = df_ml["JenjangKarir"].map(career_level)

performance_class = {"Sangat_kurang" : 1,
                     "Kurang" : 2,
                     "Biasa" : 3,
                     "Bagus" : 4,
                     "Sangat_bagus" : 5}
df_ml["PerformancePegawai"] = df_ml["PerformancePegawai"].map(performance_class)

# Working Division
engineering_div = ['Software Engineer (Back End)',
                'Software Engineer (Front End)',
                'Software Engineer (Android)',
                'Software Engineer (iOS)',
                'DevOps Engineer',
                'Software Architect',
                'Machine Learning Engineer']
data_div = ['Data Analyst', 'Data Engineer']
product_div = ['Product Manager',
                'Product Design (UX Researcher)',
                'Product Design (UI & UX)',
                'Digital Product Manager',
                'Scrum Master']

df_ml["Pekerjaan"] = df_ml["Pekerjaan"].map(lambda x: "engineering_division" if x in engineering_div else \
                                                      "data_division" if x in data_div else "product_division")

# Frequency Encoding
frequency_logic = df_ml.groupby("HiringPlatform").size()/len(df_ml)
df_ml["HiringPlatform_Encode"] = df_ml["HiringPlatform"].map(frequency_logic)
df_ml.drop(["HiringPlatform"],axis=1,inplace=True)
```

Feature Engineering

```python
feature_labelencoding = []
feature_onehot = []
feature_frequency = []

# Iteration
for i in df_categorical.columns:
    test_loop = df_categorical[i].unique().tolist()
    print(test_loop)
    # Logic of categorical transformation above
    if len(test_loop) == 2:
        print(f"{i} is Dichotomous ({len(test_loop)}): Yes or No. Strategy: Label Encoder")
        feature_labelencoding.append(i)
    elif 2 < len(test_loop) <= 6:
        print(f"{i} has 3 - 6 unique values ({len(test_loop)}). Strategy: One-hot Encoding")
        feature_onehot.append(i)
    elif len(test_loop) > 6:
        print(f"{i} has 3 - 6 unique values ({len(test_loop)}). Strategy: Frequency Encoding")
        feature_frequency.append(i)
    else:
        print(f"{i} is Other")
    print("")
```

```
['Belum_menikah', 'Menikah', 'Bercerai', 'Lainnya']
StatusPernikahan has 3 - 6 unique values (4). Strategy: One-hot Encoding

['Outsource', 'FullTime', 'Internship']
StatusKepegawaian has 3 - 6 unique values (3). Strategy: One-hot Encoding

['engineering_division', 'data_division', 'product_division']
Pekerjaan has 3 - 6 unique values (3). Strategy: One-hot Encoding

['Jakarta_Timur', 'Jakarta_Utara', 'Jakarta_Pusat', 'Jakarta_Selatan', 'Jakarta_Barat']
AsalDaerah has 3 - 6 unique values (5). Strategy: One-hot Encoding
```

Feature Transformation

The goal of feature engineering is to create variables, alter data types and formats, and clean up the data context. While feature transformation is used to change categorical and numeric data such that it is ML-ready, that is, all variables are quantitative variables (transform categorical variables into numeric with feature encoding) and have a low standard deviation value between data.

www.linkedin.com/in/fadillahakbar

Model Training

```
# Threholding Imbalanced Learning using ML Metrics

list_all_imbalanced = [tl,rus,enn,senn,stl]

imbalance_strategy = []
score_accuracy = []
score_precision = []
score_recall = []
score_auc = []
time_training = []

for imbalanced_learning in list_all_imbalanced:
    print(f"Processing {imbalanced_learning.__class__.__name__}")
    start_time = time.time()
    model = XGBClassifier(verbosity = 0,use_label_encoder = False, random_state = 123)

    # Define pipeline
    pipeline=Pipeline(steps=[("r", imbalanced_learning), ("m", model)])

    # Define evaluation procedure (here we use Repeated Stratified K-Fold CV)
    cv=RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)

    # Evaluate model
    scoring=["accuracy","precision_macro","recall_macro","roc_auc"]
    scores = cross_validate(pipeline, X_train, y_train, scoring=scoring, cv=cv, n_jobs=-1)

    imbalance_strategy.append(imbalanced_learning.__class__.__name__)
    score_accuracy.append(np.mean(scores["test_accuracy"]))
    score_precision.append(np.mean(scores["test_precision_macro"]))
    score_recall.append(np.mean(scores["test_recall_macro"]))
    score_auc.append(np.mean(scores["test_roc_auc"]))
    end_time = time.time()
    time_training.append(time.strftime('%H:%M:%S', time.gmtime(end_time - start_time)))

# Result
clear_output()
df_evaluation = pd.DataFrame({"Imbalance_Strategy":imbalance_strategy,
                              "Accuracy": score_accuracy,
                              "Precision": score_precision,
                              "Recall": score_recall,
                              "AUC": score_auc,
                              "Training_Time": time_training})
df_evaluation.sort_values(["AUC"],ascending=False)
```

	Imbalance_Strategy	Accuracy	Precision	Recall	AUC	Training_Time
3	SMOTEENN	0.932395	0.932303	0.914841	0.931276	00:00:12
4	SMOTETomek	0.949278	0.953294	0.926647	0.927692	00:00:16
1	RandomUnderSampler	0.909452	0.895626	0.898661	0.919405	00:00:13
0	TomekLinks	0.959957	0.970717	0.934286	0.914517	00:00:18
2	EditedNearestNeighbours	0.933983	0.932278	0.915883	0.913922	00:00:15

TomekLinks shows the best results, so TomekLinks will be used for the sampling strategy.

Train machine learning models in this step utilizing the recommended stages, namely Pipeline, CV, and picking the optimum Imbalanced Learning approach for existing datasets. Following that, model training will be performed on > 10 ML models in the vanilla context (with no parameter changes / default parameters), with the models being chosen based on the ML metrics employed.

```
#Initialize the classifier model
clf_1 = GaussianNB()
clf_2 = SVC(random_state=0)
clf_3 = RandomForestClassifier(random_state=0)
clf_4 = LogisticRegression()
clf_5 = DecisionTreeClassifier(random_state=0)
clf_6 = BaggingClassifier(random_state=0,base_estimator=clf_5)
clf_7 = GradientBoostingClassifier(random_state=0)
clf_8 = AdaBoostClassifier(random_state=0,base_estimator=clf_5)
clf_9 = KNeighborsClassifier()
clf_10 = MLPClassifier()
clf_11 = XGBClassifier(random_state=0)
clf_12 = LGBMClassifier(random_state=0)
clf_13 = CatBoostClassifier(random_state=0)
```

www.linkedin.com/in/fadillahakbar

Model Selection

	ML_Model	Accuracy	Precision	Recall	AUC	Training_Time
1	SVC	0.861833	0.920564	0.762302	0.933264	00:00:01
0	GaussianNB	0.958514	0.969411	0.933313	0.926422	00:00:00
6	GradientBoostingClassifier	0.950649	0.956878	0.927688	0.917867	00:00:00
11	LGBMClassifier	0.955339	0.963414	0.931022	0.917278	00:00:27
5	BaggingClassifier	0.963059	0.976624	0.936508	0.915771	00:00:00
10	XGBClassifier	0.959957	0.970717	0.934286	0.914517	00:00:16
3	LogisticRegression	0.943001	0.960116	0.904524	0.914173	00:00:00
2	RandomForestClassifier	0.963059	0.976624	0.936508	0.909248	00:00:00
12	CatBoostClassifier	0.963059	0.976624	0.936508	0.900952	00:00:04
7	AdaBoostClassifier	0.875685	0.853703	0.875050	0.875050	00:00:00
4	DecisionTreeClassifier	0.875469	0.854507	0.874772	0.874772	00:00:00
9	MLPClassifier	0.878571	0.846804	0.794196	0.860423	00:00:00
8	KNeighborsClassifier	0.837374	0.848827	0.757758	0.831040	00:00:00

Insight/Analysis

At this point, training on all vanilla ML models yielded the results that CatBoostClassifier is the best model when high Precision metrics, consistent AUC, and reasonably rapid Training Time are considered in comparison to other existing models.

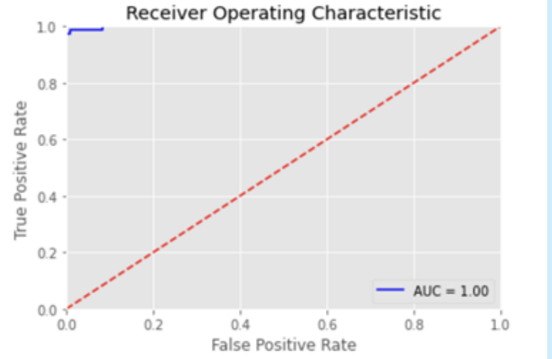
CatBoostClassifier is being considered since the training time is very short (3 seconds), and with trade-off ML metrics such as AUC, precision is not significantly different from other classifiers.

This model will then be hyperparameter tuned to achieve the best results.

Hyperparameter Tuning

At this stage, hyperparameter tuning is performed using Bayesian search through the Hyperopt library, and the best parameter tuning results are used again to perform model training so that the results can be seen as in the classification report and performance metrics, where the value of the existing metrics2 is stable at 96%, then the precision metrics that are primarily used in this ML model as business metrics have a precision category 0 of 96% and category 1 of 96%.

```
roc_plot(catboost_model,X_train,y_train)
```




```
roc_plot(catboost_model,X_test,y_test)
```

The strong AUC scores on the train and test data indicate that the model has been hyperparameter tuned and is ready to be used for the purposes of analysis and automation of churn determination for businesses.

```
from hyperopt import space_eval

# Best_params of the best model
best_params = space_eval(space, best_classifier)
best_params
```

```
{'learning_rate': 0.010132475113471593,
 'max_depth': 3.0,
 'n_estimators': 480.0}
```

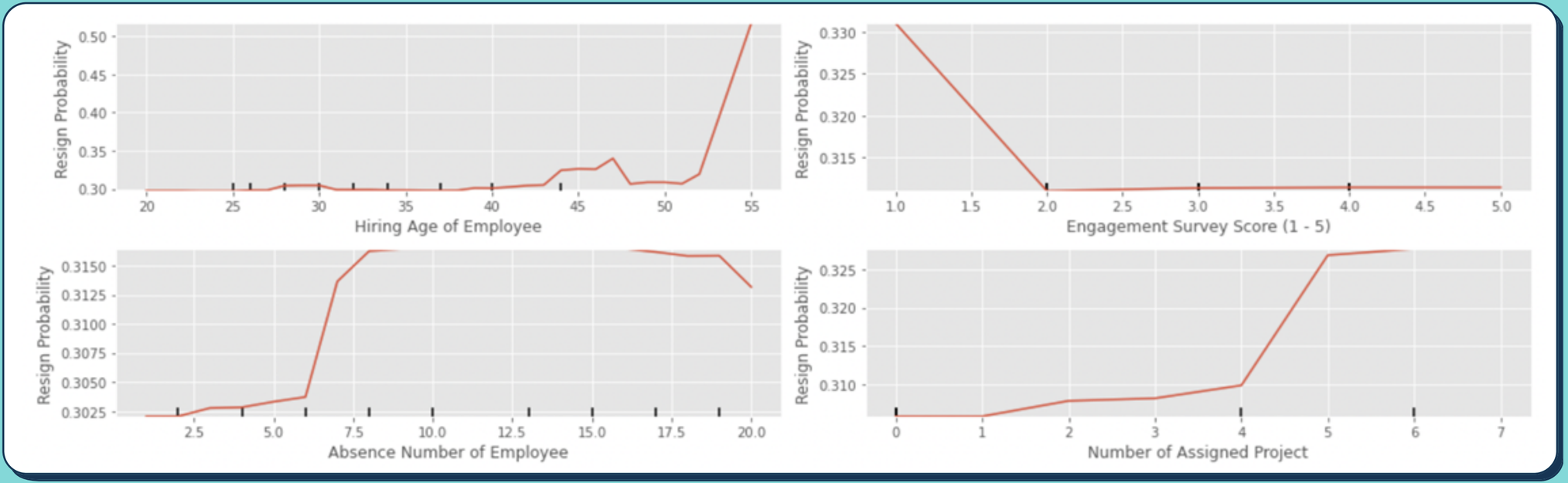
```
display_model_performance_metrics(true_labels=y_test,
                                  predicted_labels=y_pred,
                                  classes=[0,1])
```

```
Model Performance metrics:
------------------------------
Accuracy: 0.9636
Precision: 0.9653
Recall: 0.9636
F1 Score: 0.9624

Model Classification report:
------------------------------
              precision    recall  f1-score   support

           0       0.96      1.00      0.98        43
           1       1.00      0.83      0.91        12

    accuracy                           0.96        55
   macro avg       0.98      0.92      0.94        55
weighted avg       0.97      0.96      0.96        55


Prediction Confusion Matrix:
------------------------------
           Predicted:
                         0   1
Actual: 0               43   0
        1                2  10
```

Presenting Machine Learning Products to the Business Users

Present machine learning solutions that use interpretable and/or explainable AI methodologies, such as gaining insight through feature importance and determining the variables that most influence people to resign, etc.

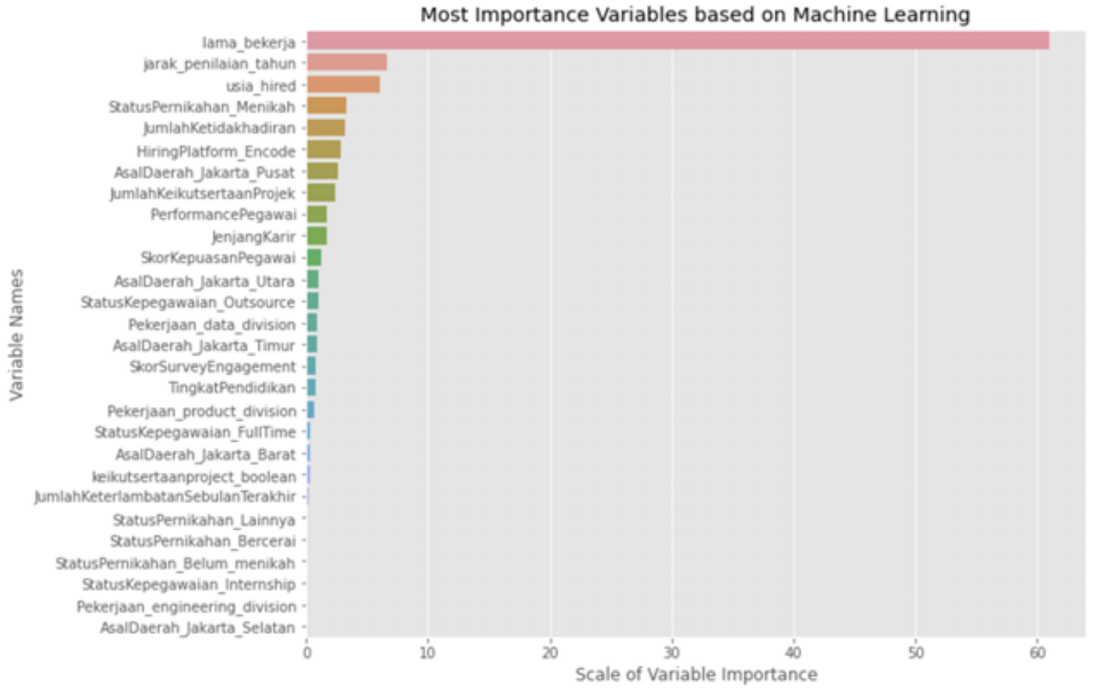
Model Selection

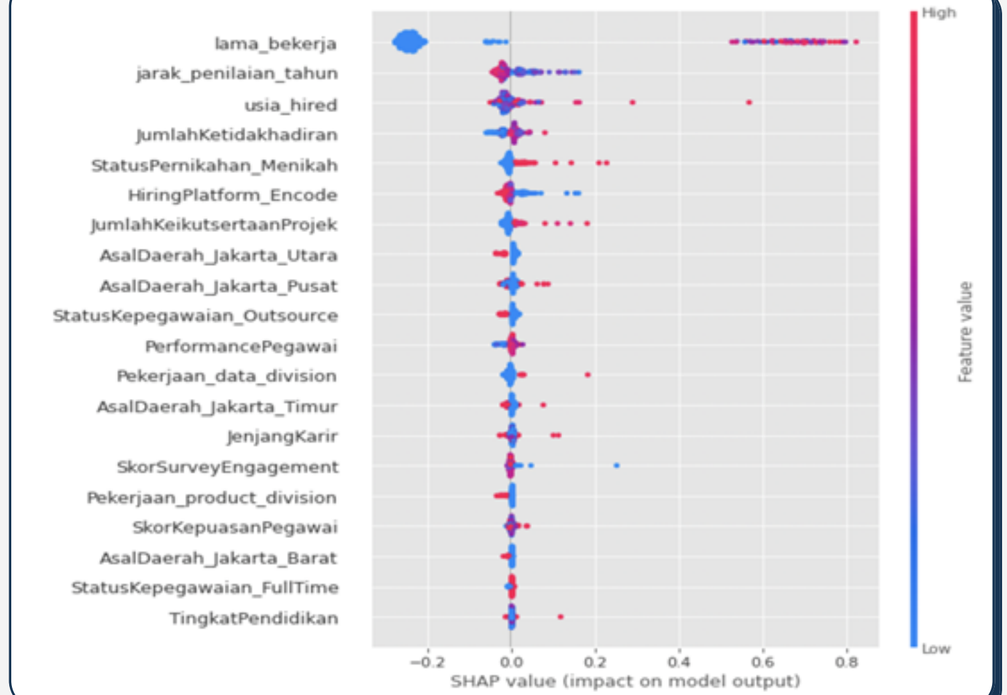


1. Employees over the age of 45 have a 5 to 20 percent higher chance of resigning than younger employees. Especially if the employee is now over the age of 52, the likelihood of quitting is very significant (>50%).
2. Employees who write a score of 1 on the engagement survey have a higher chance of resigning than those who give a score of 2 to 5.
3. Employees who are absent for more than 5 days are more likely to resign than those who are absent for less than 5 days.
4. Based on the number of projects in which employees participate, the more often they are involved in projects that turn out to be negative for them, the greater their odds of resigning, especially if they participate in more than four projects.

www.linkedin.com/in/fadillahakbar

Model Selection





1. The longer an employee works, the more likely he or she will resign.
2. The older the hired person, the more likely they are to resign.
3. The greater the absenteeism, the more likely resignation.
4. Employees hired through online-web-based, other websites are more likely to resign than those hired through Linkedin, Indeed, and Google.
5. The more the employee's involvement in the project, the more likely they are to resign.

Thank you!

For more details, you can check out the jupyter notebook here.
The dataset can be seen here.

www.linkedin.com/in/fadillahakbar

Resource Page

Use these icons and illustrations in your Canva Presentation. Happy designing!

Resource Page

Use these icons and illustrations in your Canva Presentation. Happy designing!

Resource Page

Find the magic and fun in presenting with Canva Presentations. Press the following keys while on Present mode!

B for blur

C for confetti

D for a drumroll

O for bubbles

Q for quiet

X to close

Any number from 0-9 for a timer