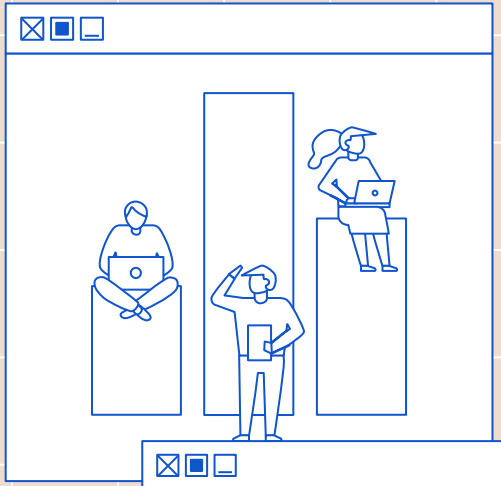
# Analisis Akurasi dan Interpretasi **Model XGBoost** untuk **Pricing** Asuransi Kompensasi Pekerja

By Jonathan Stanley

Xtreme Gradient Boosting



# 01

# PENDAHULUAN

# LATAR BELAKANG

Semua pekerjaan di sebagian besar negara wajib memiliki asuransi kompensasi pekerja. Asuransi kompensasi pekerja meliputi biaya pengobatan, kehilangan upah dan tunjangan lainnya jika seorang karyawan sakit saat bekerja.

Setiap pekerja tentu memiliki tingkat risiko yang berbeda-beda. Penilaian tingkat risiko klaim pemohon asuransi merupakan bagian penting dalam asuransi, sehingga perlu untuk diklasifikasikan. Penentuan tingkat risiko klaim pada asuransi didasarkan pada data historis pemohon. Pengajuan untuk menjadi anggota suatu asuransi membutuhkan waktu yang tidak singkat.

# LATAR BELAKANG

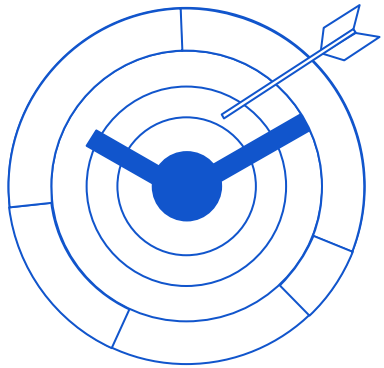
XGBoost

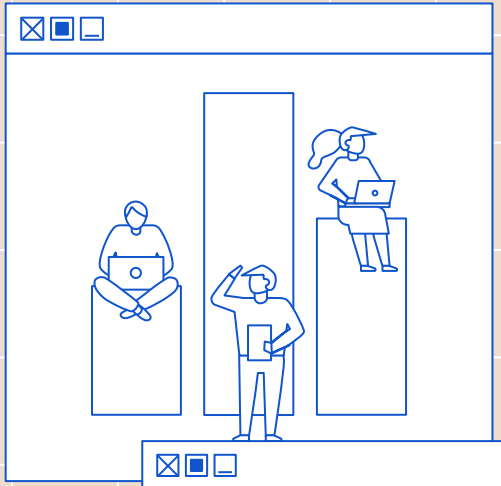
XGBoost dipilih sebagai metode utama untuk penelitian ini dikarenakan performanya yang cenderung lebih baik dibandingkan model lain untuk berbagai masalah.

# TUJUAN PENELITIAN

1) Mengevaluasi performa statistik model XGBoost terhadap masalah pricing asuransi kompensasi pekerja

2) Mendapatkan hasil interpretasi dari model XGBoost terhadap masalah pricing asuransi kompensasi pekerja





# 02

# METODE

# DEFINISI XGBOOST

Metode XGBoost merupakan pengembangan dari metode Gradient Boosting. Gradient boosting merupakan algoritma yang dapat menemukan solusi yang optimal untuk masalah regresi, dan klasifikasi. Konsep dasar dari algoritma ini adalah menyesuaikan parameter pembelajaran secara berulang untuk meminimalkan loss function (mekanisme evaluasi atas model). XGBoost menggunakan model yang lebih teratur untuk membangun struktur pohon regresi, sehingga memberikan kinerja yang lebih baik, dan mengurangi kompleksitas model untuk menghindari overfitting. Berbeda dengan Gradient Boosting biasa, XGBoost menggunakan metode aproksimasi turunan kedua dan fungsi regularisasi L1 dan L2 dalam proses meminimalkan loss function.

# TAHAPAN METODE PENELITIAN

## Pemrosesan Data

Sumber data publik mengenai Asuransi Kompensasi Pekerja

## Modelling

Proses Modelling XGBoost

## Evaluasi

Evaluasi performa model menggunakan metrik evaluasi

# Data Processing

**Data Collecting :** Menggunakan dataset publik yang tersedia di Kaggle, yaitu dataset Actuarial loss prediction mengenai asuransi kompensasi pekerja.

**Data Understanding :** Memahami setiap variabel pada dataset serta mendapatkan pemahaman yang lebih lanjut untuk menarik insight dengan menggali data lebih dalam menggunakan visualisasi, dan EDA.

**Data Preprocessing :** Melakukan filter sehingga data yang tidak relevan dihilangkan, serta melakukan pembersihan data untuk missing values, invalid data, unstructured data, imbalance data, dan masalah data lainnya.

**Data Transformation :** Merubah data menjadi bentuk yang terstruktur sebelum modeling, dimana pada regresi XGBoost data akan diubah menjadi bentuk numerik menggunakan *one hot encoding*.

# Data Modelling

**Data Splitting :** Membagi data menjadi dua bagian, yaitu data training dan data testing.

**Model Building :** Membuat model XGBoost menggunakan pilihan model regresi menggunakan *XGBRegressor*

**Optimisation :** Mencari hyperparameter yang terbaik untuk model menggunakan Bayes Search dan juga K-fold Cross Validation

**Model Prediction:** Membuat prediksi menggunakan Model XGBoost dengan menggunakan testing data

# Result Evaluation

**A) Evaluation Metrics**

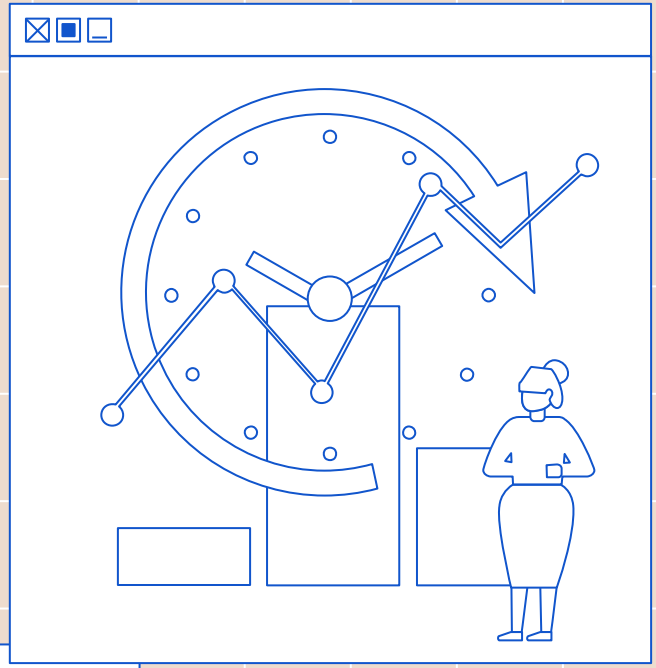
- **Mean Absolute Error (MAE)**
- **Root Mean Squared Error(RMSE)**
- **R-Squared (R^2)**

**B) Model Interpretation**

Melakukan interpretasi terhadap model XGBoost untuk pricing asuransi kompensasi pekerja dengan cara mengevaluasi seberapa pentingnya variabel dalam mempengaruhi hasil prediksi (**Feature Importance**)

# 03

# SIMULASI DAN ANALISIS HASIL



Simulasi Pembuatan Model XGBoost

# DATASET

Dataset yang digunakan adalah ‘**Actuarial loss prediction**’ , sebuah dataset publik yang diperoleh melalui Kaggle

**Dataset ini mencakup 90.000 kebijakan asuransi kompensasi pekerja yang realistis, yang semuanya mengalami kecelakaan. Untuk setiap catatan, terdapat informasi demografis dan terkait pekerja, serta deskripsi teks mengenai kecelakaan tersebut.**

# DATASET INFORMATION

**Data fields (15 fields)**

- ClaimNumber: Identifikasi unik kebijakan
- DateTimeOfAccident: Tanggal dan waktu kecelakaan
- DateReported: Tanggal pelaporan kecelakaan
- Age: Usia pekerja
- Gender: Jenis kelamin pekerja
- MaritalStatus: Status pernikahan pekerja. (M)enikah, (S)ingle, (U)nknown.
- DependentChildren: Jumlah anak tanggungan
- DependentsOther: Jumlah tanggungan lainnya selain anak-anak
- WeeklyWages: Total upah mingguan
- PartTimeFullTime: Biner (P) atau (F)
- HoursWorkedPerWeek: Total jam kerja per minggu
- DaysWorkedPerWeek: Jumlah hari kerja per minggu
- ClaimDescription: Deskripsi bebas teks tentang klaim
- InitialIncurredClaimCost: Estimasi awal oleh perusahaan asuransi mengenai biaya klaim
- UltimateIncurredClaimCost: Total pembayaran klaim oleh perusahaan asuransi.

# DATA READING

## train.csv

- Dataset untuk training model dengan lebih dari 54.000 insurance policies

## Mengubah Fitur DateTime

```
#pengubahan tipe fitur DateTime & seleksi data
df['DateTimeOfAccident'] = pd.to_datetime(df['DateTimeOfAccident'])
df['DateReported'] = pd.to_datetime(df['DateReported'])
df['YearReported'] = df['DateReported'].dt.year
```

# DATA UNDERSTANDING

- **5 Baris Pertama dari dataframe**

| ClaimNumber | WC8285054 | WC6982224 | WC5481426 | WC9775968 | WC6828422 |
|---|---|---|---|---|---|
| DateTimeOfAccident | 2002-04-09 07:00:00+00:00 | 1999-01-07 11:00:00+00:00 | 1996-03-25 00:00:00+00:00 | 2005-06-22 13:00:00+00:00 | 1999-06-21 11:00:00+00:00 |
| DateReported | 2002-07-05 00:00:00+00:00 | 1999-01-20 00:00:00+00:00 | 1996-04-14 00:00:00+00:00 | 2005-07-22 00:00:00+00:00 | 1999-09-09 00:00:00+00:00 |
| Age | 48 | 43 | 30 | 41 | 50 |
| Gender | M | F | M | M | M |
| MaritalStatus | M | M | U | S | M |
| DependentChildren | 0 | 0 | 0 | 0 | 0 |
| DependentsOther | 0 | 0 | 0 | 0 | 0 |
| WeeklyWages | 500.0 | 509.34 | 709.1 | 555.46 | 200.0 |
| PartTimeFullTime | F | F | F | F | F |
| HoursWorkedPerWeek | 38.0 | 37.5 | 38.0 | 38.0 | 38.0 |
| DaysWorkedPerWeek | 5 | 5 | 5 | 5 | 5 |
| ClaimDescription | LIFTING TYRE INJURY TO RIGHT ARM AND WRIST INJURY | STEPPED AROUND CRATES AND TRUCK TRAY FRACTURE ... | CUT ON SHARP EDGE CUT LEFT THUMB | DIGGING LOWER BACK LOWER BACK STRAIN | STRUCK HEAD ON HEAD LACERATED HEAD |
| InitialIncurredCalimsCost | 1500 | 5500 | 1700 | 15000 | 500 |
| UltimateIncurredClaimCost | 4748.203388 | 6326.285819 | 2293.949087 | 17786.48717 | 598.762315 |
| YearReported | 2002 | 1999 | 1996 | 2005 | 1999 |

# DATA UNDERSTANDING

- **Statistical Descriptive dari data numerik pada dataframe**

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| Age | 45291.0 | 34.009207 | 12.159998 | 13.000000 | 24.00000 | 32.000000 | 43.000000 | 81.000 |
| DependentChildren | 45291.0 | 0.079861 | 0.418868 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 9.000 |
| DependentsOther | 45291.0 | 0.007397 | 0.090693 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 3.000 |
| WeeklyWages | 45291.0 | 428.014364 | 260.023794 | 1.000000 | 200.00000 | 412.770000 | 521.180000 | 7497.000 |
| HoursWorkedPerWeek | 45291.0 | 37.569711 | 12.452778 | 0.000000 | 38.00000 | 38.000000 | 40.000000 | 640.000 |
| DaysWorkedPerWeek | 45291.0 | 4.894063 | 0.581220 | 1.000000 | 5.00000 | 5.000000 | 5.000000 | 7.000 |
| InitialIncurredCalimsCost | 45291.0 | 8753.829922 | 22102.047855 | 1.000000 | 756.00000 | 2500.000000 | 10000.000000 | 2000000.000 |
| UltimateIncurredClaimCost | 45291.0 | 11947.669219 | 35420.756715 | 121.886805 | 945.02231 | 3723.166477 | 9116.191048 | 4027135.935 |
| YearReported | 45291.0 | 1998.066106 | 4.352081 | 1991.000000 | 1994.00000 | 1998.000000 | 2002.000000 | 2006.000 |

# DATA PREPROCESSING

## Missing Values

- Mencari kolom yang memiliki missing values

```
# Hitung missing values
df.isnull().sum()
```

✓ 0.0s

```
DateTimeOfAccident            0
DateReported                  0
Age                           0
Gender                        0
MaritalStatus                28
DependentChildren             0
DependentsOther               0
WeeklyWages                   0
PartTimeFullTime              0
HoursWorkedPerWeek            0
DaysWorkedPerWeek             0
ClaimDescription              0
InitialIncurredCalimsCost     0
UltimateIncurredClaimCost     0
YearReported                  0
dtype: int64
```

**Terdapat missing value pada field ‘MaritalStatus’ yaitu sebanyak 28 values pada baris yang berbeda.**

# DATA PREPROCESSING

## Feature Selection

```
#penghapusan data dengan nilai fitur tidak realistis
df = df.loc[(df['HoursWorkedPerWeek'] <= 168)&(df['HoursWorkedPerWeek']>0)&
           (df['ReportingDelay']>=0)]
```

```
#penghapusan unknown value

df.dropna(inplace=True)
df = df.loc[(df['Gender'] != 'U')&(df['MaritalStatus'] != 'U')]
```

# DATA PREPROCESSING

## Missing Values

- ‘MartialStatus’ akan di analisa lebih lanjut dan missing values akan ditangani

'MartialStatus' memiliki value : (M)arried, (S)ingle, (U)nknown. Missing values pada kolom ini akan disubtitusikan dengan nilai **U .**

```
# Mengisi missing values dengan status U = Unknown
df['MaritalStatus']= df['MaritalStatus'].fillna('U')
```

✓ 0.0s

| MaritalStatus | mean | median | count |
|---|---|---|---|
| M | 13028.654077 | 4522.489306 | 18654 |
| S | 9857.401085 | 2486.967424 | 21926 |
| U | 17466.998886 | 6469.310541 | 4683 |

# DATA PREPROCESSING

## Feature Engineering

- Membuat field baru ‘ReportingDelay’ yaitu jarak waktu dari waktu kejadian di laporkan dan waktu dari kejadian.
- Menghapus Field ‘ClaimDescription’ karena valuenya tidak relevan untuk prediksi.

```
#pengubahan tipe fitur DateTime

df['DateTimeOfAccident'] = pd.to_numeric(df['DateTimeOfAccident'])
df['DateReported'] = pd.to_numeric(df['DateReported'])

#pemilihan dan rekayasa fitur
df['ReportingDelay'] = (df['DateReported'] - df['DateTimeOfAccident'])
df.drop(columns=['ClaimDescription'], inplace=True)
```

# DATA TRANSFORMATION

- Mengganti field dengan value kategorik dengan nilai dummy menggunakan One Hot Encoding
- Mendefinisikan target variabel
- Melakukan Normalisasi pada data

```
#One Hot Encoding
category = pd.get_dummies(df[['Gender','MaritalStatus','PartTimeFullTime']],
                          drop_first=True)
#Target variabel
label = df[['UltimateIncurredClaimCost']]
df.drop(columns=['DateReported','Gender','MaritalStatus','PartTimeFullTime',
                 'UltimateIncurredClaimCost','YearReported'], inplace=True)
df = pd.concat([df, category], axis=1)

#normalisasi
scaler = MinMaxScaler()
df[df.columns] = scaler.fit_transform(df)
df
```

# DATA MODELLING

## XGBRegressor

Fungsi gradient boosting dengan algoritma yang digunakan untuk regresi

## Bayesian Optimization

Teknik optimasi untuk mencari hyperparameters yang terbaik untuk performa model

# Hyperparameters

- alpha : Untuk regularisasi L1 **(0.01 – 10)**
- learning_rate : Ukuran langkah setiap iterasi boosting **(0.01 – 0.5)**
- max_depth : Maximum depth dari setiap decision tree **(2 – 10)**
- n_estimators : Jumlah iterasi boosting **(50 – 1000)**
- min_child_weight : Jumlah minimum dari beban yang dibutuhkan pada node **(1 – 10)**
- subsample : Proporsi dari data pelatihan **(0.5 – 1.0)**
- colsample_bytree : Proporsi dari  fitur yang dipilih secara acak **(0.1 – 1.0)**
- gamma : Minimum loss reduction yang dibutuhkan **(0 – 10)**

```
# Mendefinisikan Jangkauan searching untuk hyperparameter
params = {
    "alpha": Real(0.01, 10, prior='log-uniform'),
    "learning_rate": Real(0.01, 0.5, prior='uniform'),
    "max_depth": Integer(2, 10),
    "n_estimators": Integer(50, 1000),
    "min_child_weight": Real(1, 10),
    "subsample": Real(0.5, 1.0, prior='uniform'),
    "colsample_bytree": Real(0.1, 1.0, prior='uniform'),
    "gamma": Real(0, 10)
}
```

# Evaluation Metrics

- **Root Mean Squared Error**

Mengukur akar dari rata-rata kuadrat kesalahan antara nilai prediksi dan nilai aktual pada skala yang sama. Semakin rendah nilai RMSE, semakin baik performa model

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

- **Mean Absolute Error**

Mengukur rata-rata kesalahan absolut antara nilai prediksi dan nilai aktual pada skala yang sama. Semakin rendah nilai MAE, semakin baik performa model

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

# Evaluation Metrics

- **R - Squared ($R^2$)**

  Mengukur seberapa dekat nilai prediksi dengan nilai aktual dalam skala 0 – 1.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

  * Dimana :

  n = banyaknya data (observasi)

  yi = nilai observasi ke-i

  $\hat{y}i$ = nilai prediksi observasi ke-i

  $\bar{y}$ = rata - rata nilai observasi

# DATA MODELLING

## Modelling Process

Modelling akan dilakukan sebanyak 5 pengulangan menggunakan data yang di split secara berbeda dengan random_state 0 sampai 4 dan akan diambil nilai rata-rata dari setiap metrik evaluasi

## 4-fold Cross Validation

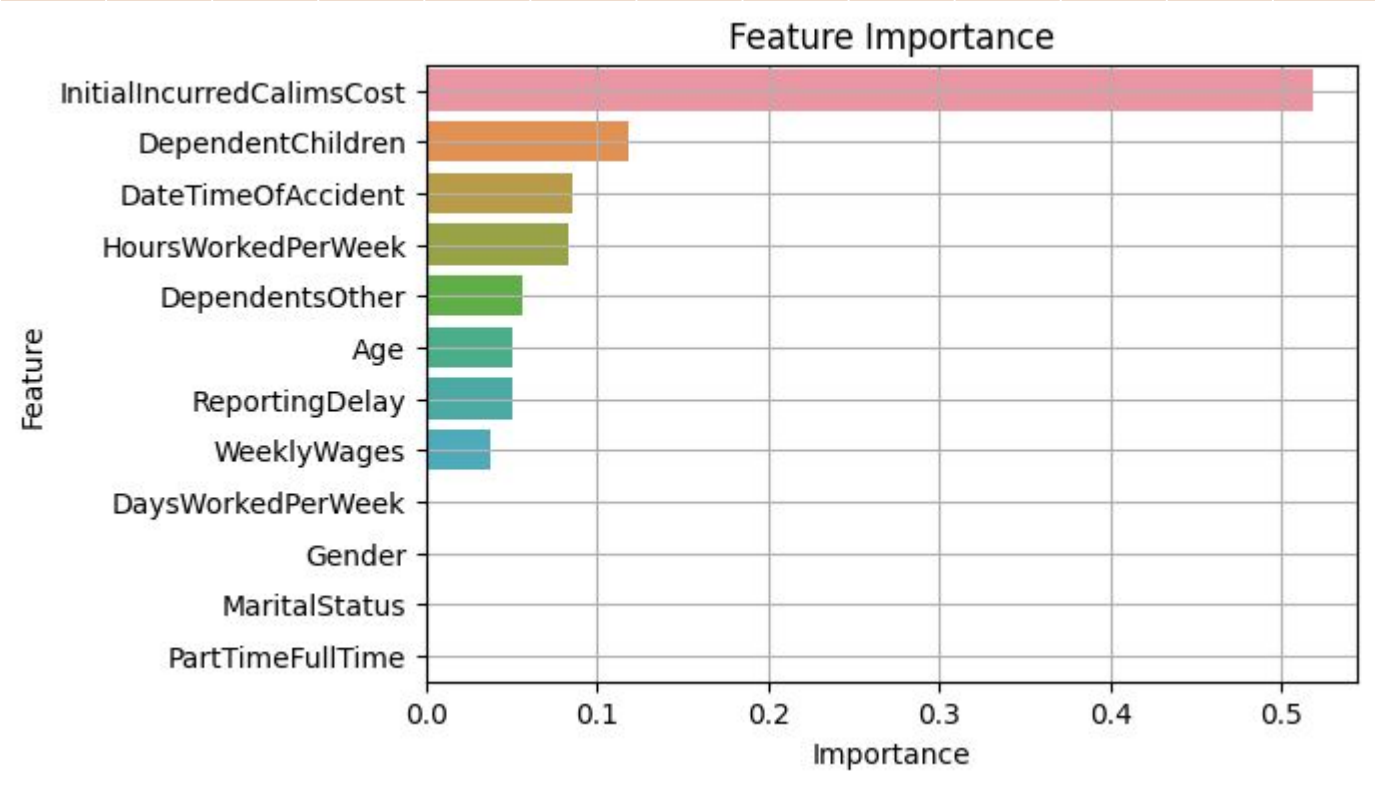
Pada setiap iterasi, data training akan dibagi menjadi 4 folds, dimana model akan dilatih pada 3 folds yang telah dibuat, dan dievaluasi pada 1 fold yang tersisa untuk performanya.

# Evaluation Result

- Nilai rata-rata hasil evaluasi dari 5 iterasi

| Evaluation Metrics | RMSE | MAE | R^2 |
|---|---|---|---|
| Mean | 23,898.08 | 7,567.58 | 0.32167 |

# Feature Importance



# Feature Importance

- Fitur yang paling besar pengaruhnya terhadap hasil prediksi model adalah ‘InitialIncurredCalimsCost’ yang melebihi tingkat importance 0.5 dari total keseluruhannya, diikuti oleh ‘Dependent Children’ dengan tingkat importance sebesar kurang lebih 0.1
- Variabel yang paling kecil pengaruhnya terhadap hasil prediksi model adalah ‘PartTimeFullTime’.

# Summary

- Model XGBoost sudah cukup baik dalam memprediksi pembayaran klaim oleh perusahaan asuransi berdasarkan metrik evaluasi RMSE, MAE, dan juga R-squared untuk digunakan sebagai metode *pricing* asuransi.
- Akan lebih baik jika model XGBoost dibandingkan dengan metode umum untuk pricing seperti Generalized Linear Model.
- Metode XGBoost telah berhasil diinterpretasi melalui feature importance untuk menjelaskan seberapa besar kontribusi suatu fitur dalam mengambil keputusan.

# References

- Dataset : https://www.kaggle.com/c/actuarial-loss-estimation
- Github Repository : https://github.com/Jonathans35/Portofolio/tree/main

# THANK YOU

**Jonathan Stanley**

jonathanstanleyofficial@gmail.com

082112426652