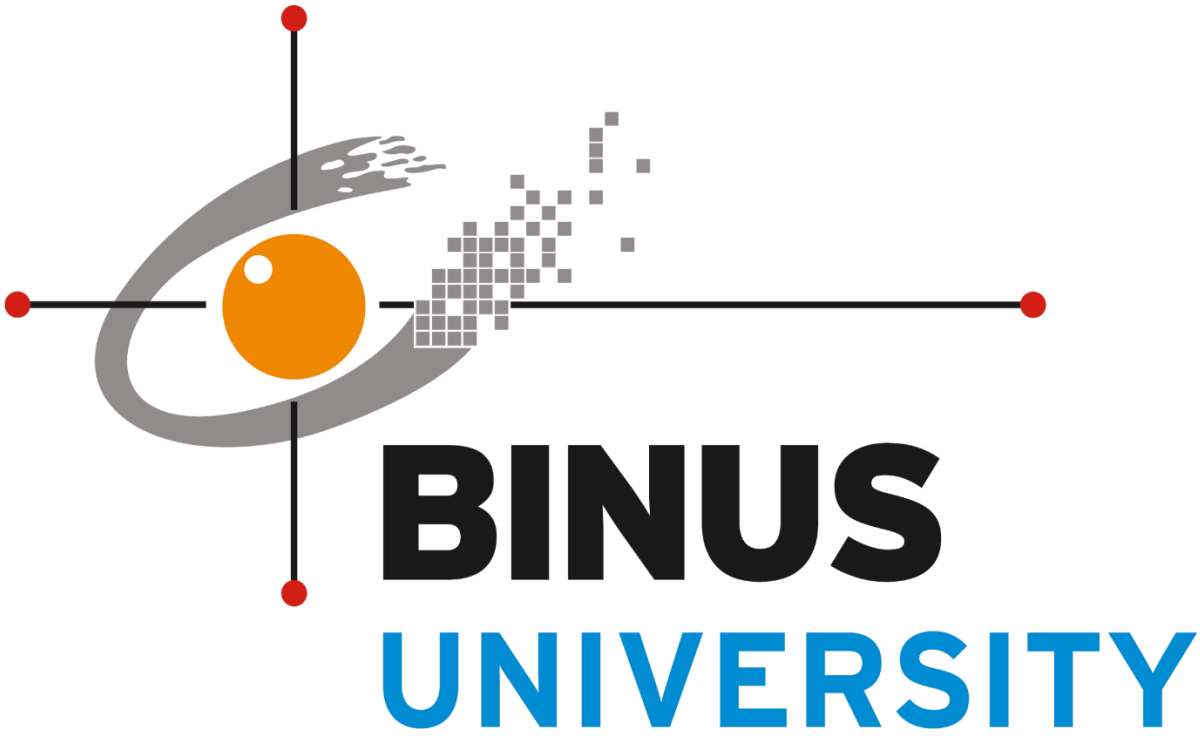
STUDENT PERFORMANCE PREDICTION

Brandon Tumiwa - 2802508224

Justine Ria Jingga - 2802536783

Raka Priyahita Pramudito - 2802544500

Stanley Angkasa - 2802550642



LA84 - Machine Learning Operations

Jurusan Artificial Intelligence Semester 3

School of Computer Science

Abstrak

Prediksi performa akademik siswa merupakan salah satu penerapan machine learning yang penting dalam bidang pendidikan untuk membantu memahami faktor-faktor yang memengaruhi hasil belajar. Pada penelitian ini dikembangkan sebuah model regresi untuk memprediksi nilai ujian siswa berdasarkan berbagai fitur akademik dan non-akademik, seperti jam belajar, kehadiran, kondisi keluarga, serta lingkungan belajar. Proses pengembangan model diawali dengan Exploratory Data Analysis (EDA) untuk memahami karakteristik data dan pola hubungan antar variabel, dilanjutkan dengan tahap preprocessing yang mencakup penanganan missing values, encoding fitur kategorikal, dan normalisasi fitur numerik. Model XGBoost dipilih sebagai model utama karena kemampuannya dalam menangani data tabular dan hubungan non-linear. Hasil evaluasi menunjukkan bahwa model memiliki performa yang baik dan stabil dengan tingkat kesalahan prediksi yang rendah. Model selanjutnya diintegrasikan ke dalam aplikasi web berbasis Streamlit dan dideploy ke cloud sehingga dapat digunakan secara real time.

Kata kunci: *machine learning, regresi, student performance, XGBoost, deployment*

BAB I
PENDAHULUAN

A. LATAR BELAKANG

Pendidikan merupakan salah satu faktor utama dalam pembangunan manusia. Prestasi akademik siswa dipengaruhi oleh banyak aspek, mulai dari kebiasaan belajar, dukungan keluarga, hingga kondisi sekolah. Dengan berkembangnya teknologi *Machine Learning*, kita dapat menganalisis faktor-faktor tersebut secara sistematis untuk memahami pola yang memengaruhi hasil ujian. Proyek ini bertujuan untuk mengintegrasikan pendekatan MLOps agar pipeline analisis dan prediksi dapat berjalan otomatis, terukur, dan berkelanjutan.

B. TUJUAN *PROJECT*

- Mengembangkan model *regresi* untuk memprediksi nilai ujian akhir siswa.
- Mengidentifikasi faktor-faktor yang paling berpengaruh terhadap performa akademik.
- Menerapkan prinsip MLOps untuk memastikan proses pengolahan data, pelatihan model, dan deployment berjalan efisien serta dapat di-*scale*.

C. PERMASALAHAN YANG INGIN DISELESAIKAN

- Bagaimana memprediksi nilai ujian akhir siswa berdasarkan faktor-faktor akademik dan non-akademik.
- Bagaimana mengelola pipeline ML secara berkelanjutan dengan MLOps.
- Bagaimana menginterpretasikan fitur-fitur yang paling signifikan terhadap performa siswa.

BAB II
ANALISIS DATASET

Dataset ini berasal dari Kaggle dengan judul *Student Performance Factors*. Data mencakup berbagai aspek yang memengaruhi performa akademik siswa dari beberapa sekolah private dan public. Tujuannya adalah memberikan gambaran komprehensif tentang faktor-faktor yang memengaruhi nilai ujian.

Dataset memiliki 6600 baris dengan 20 kolom (19 fitur + 1 target):
Fitur numerik terdiri dari Hours_Studied, Attendance, Sleep_Hours, Previous_Score, Tutoring_Sessions, Physical_Activity
Fitur kategorikal tediri dari Parental_Involvement, Access_to_Resources, Extracurricular_Activities, Motivation_Level, Internet_Access, Family_Income, Teacher_Quality, School_Type, Peer_Influence, Learning_Disabilities , Parental_Education_Level, Distance_from_Home, Gender

2.1 Analisis Fitur Numerik

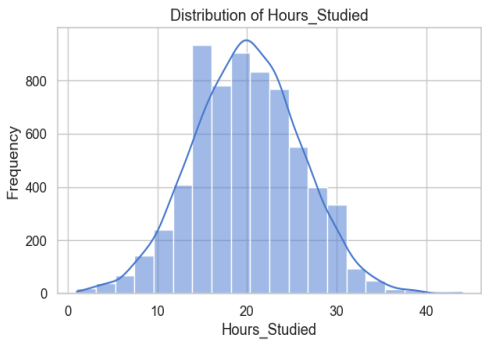
2.1.1 Statistik Deskriptif Fitur Numerik

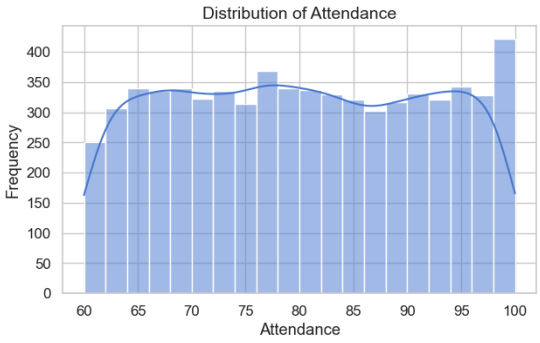
Analisis ini menggunakan fungsi describe dari pandas, dan bertujuan untuk memahami range nilai fitur numerik dan variasi nilai target

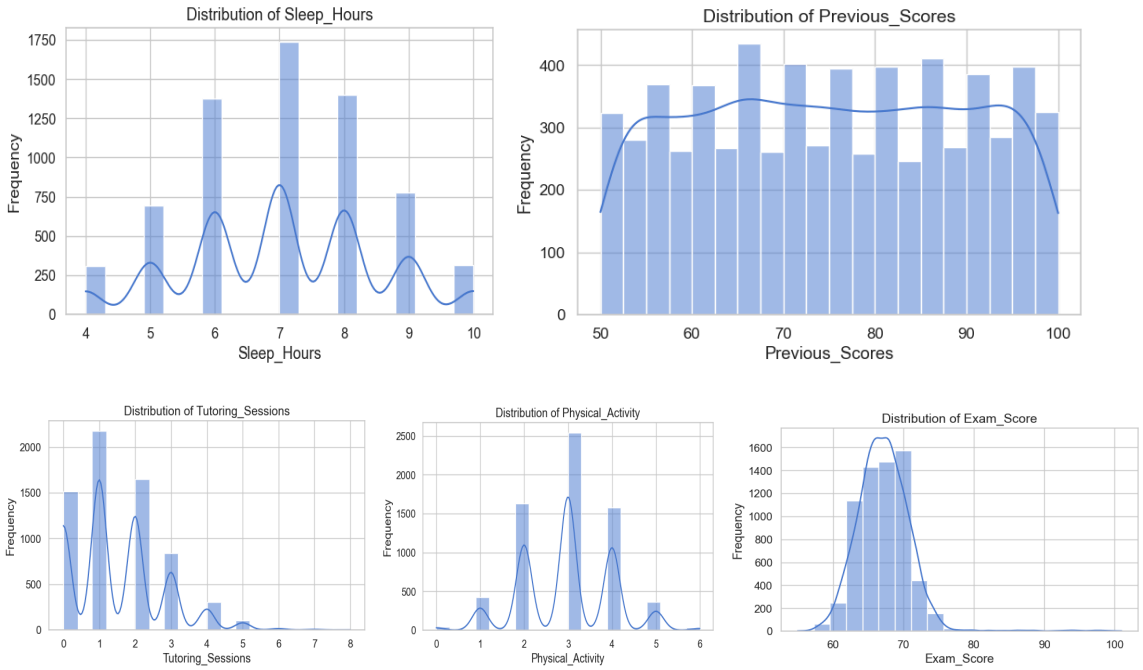
Secara umum, dataset memiliki tipe data yang konsisten dan range nilai yang wajar, misalnya Exam_Score berkisar antara **55 hingga 101**, dengan rata-rata sekitar **67 poin.**

2.1.2 Distribusi Fitur Numerik

Kami menggunakan histogram dan plot digunakan untuk melihat bentuk distribusi fitur numerik seperti *Hours_Studied, Attendance, Sleep_Hours, Previous_Scores*, dan fitur lainnya. Berikut visualisasi untuk distribusi tiap fitur numerik.





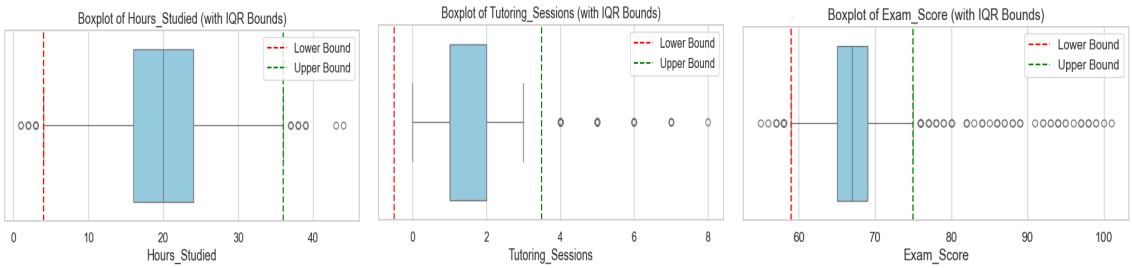


Penjelasan distribusi:

- Untuk fitur *Hours_Studied, Sleep_Hours, dan Physical Activity*, distribusinya relatif merata dan cenderung mengikuti distribusi normal
- Untuk fitur *Attendance* dan *Previous_Scores,* distribusinya relatif seimbang dan tidak ada nilai yang menonjol/dominan dengan kuat, distribusi cenderung mengikuti *Uniform Distribution*
- Untuk fitur *Tutoring_Sessions dan Exam_Scores* (fitur target), distribusi memiliki skewness ke arah kiri (*left-skewed*). Di tutoring_sessions, kebanyakan siswa memiliki sesi yang sedikit (di range 1-3), dimana dari 4 dan seterusnya terus menurun. Di Exam_Scores, mayoritas siswa memiliki nilai di range mendekati mean yaitu di 60-70, sehingga setelah 70, frequency/kemunculan turun dengan sangat drastis. Ini menunjukkan range nilai (exam_scores) di dataset cukup sempit (low variance).

2.1.3 Analisis Outlier dalam Fitur Numerik

Analisis outlier dilakukan menggunakan metode Interquartile Range (IQR) dan divisualisasikan melalui boxplot untuk setiap fitur numerik. Proses ini bertujuan untuk mengidentifikasi nilai-nilai ekstrem yang berada di luar batas bawah ($Q1 - 1.5 \times IQR$) dan batas atas ($Q3 + 1.5 \times IQR$). Dengan menghitung Q1, Q3, serta IQR untuk setiap fitur, dapat diketahui apakah terdapat pengamatan yang secara signifikan berbeda dari mayoritas data.



Perhitungan outlier

Perhitungan dilakukan dengan mencari nilai quartil 1 dan quartil 3, lalu menggunakan Q3-Q1 untuk menghitung IQR (*interquartile range)*, yang kemudian dipakai untuk menghitung batas atas (*upper bound)* dan batas bawah (*lower bound*) dari distribusi fitur tersebut

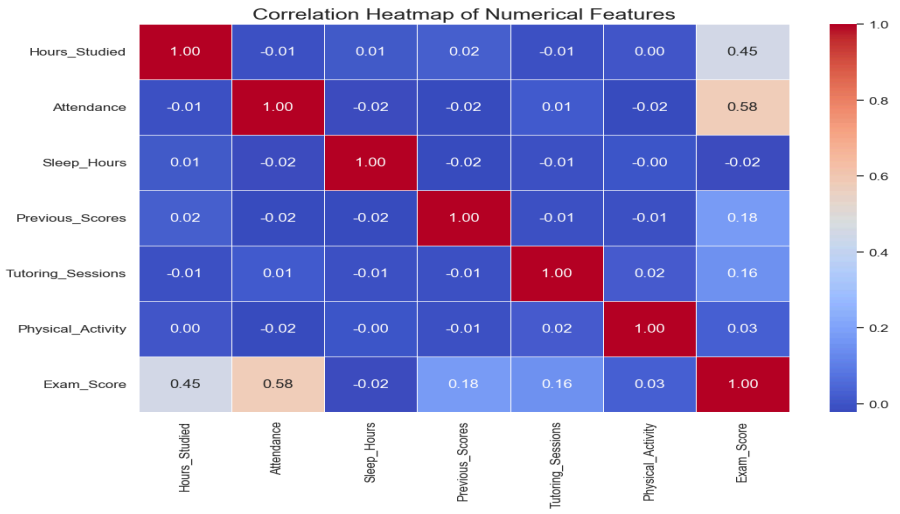
Jumlah outlier yang didapatkan dari perhitungan :

- Hours_Studied: **43 rows**
- Attendance, Sleep_Hours, Previous_Scores, Physical_Activities: **tidak ada outlier**
- Tutoring_Sessions : **430 rows** (sesuai histogram, nilai yang di bagian kanan dihitung sebagai outlier). Ini bukan outlier yang "salah", tetapi hanya distribusi yang skewed.
- Exam_Scores : **104 rows,** hal ini menunjukkan bahwa memang nilai menumpuk di sekitar mean.

Setelah dianalisis lebih jauh, outlier tersebut merupakan nilai yang masih masuk akal secara domain dan mencerminkan variasi alami antar siswa. Misalnya, jumlah *Tutoring_Sessions* yang tinggi memang jarang terjadi tetapi bukan merupakan kesalahan data. Begitu pula dengan nilai ujian yang berada di atas rentang 75–80, yang secara statistik jarang namun valid secara akademis. Karena outlier bersifat wajar dan tidak ekstrem, serta tidak menunjukkan adanya kesalahan pencatatan data, keputusan diambil untuk mempertahankan seluruh outlier dan tidak melakukan proses penghapusan atau transformasi. Hal ini juga mempertahankan integritas distribusi asli data dan mencegah hilangnya informasi penting yang dapat berkontribusi pada pemodelan regresi.

2.1.4 Analisis Korelasi

Analisis korelasi Pearson digunakan untuk mengukur hubungan linear antara fitur numerik dan variabel target *Exam_Score*.



Hasil Analisis

Hasil visualisasi heatmap menunjukkan bahwa beberapa fitur numerik memiliki korelasi yang cukup kuat terhadap nilai ujian. Fitur dengan korelasi tertinggi adalah *Attendance* (0.58), diikuti oleh *Hours_Studied* (0.45), yang mengindikasikan bahwa tingkat kehadiran dan jumlah waktu belajar merupakan dua faktor yang paling berpengaruh terhadap performa siswa. Kedua nilai korelasi ini dapat dikategorikan sebagai hubungan linear moderat hingga kuat.

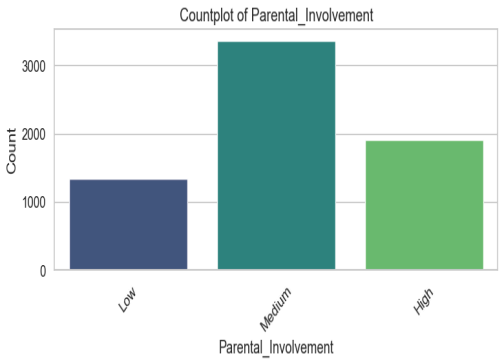
Sementara itu, fitur seperti *Previous_Scores* (0.17), *Tutoring_Sessions* (0.16), dan *Physical_Activity* (0.03) memiliki hubungan yang jauh lebih lemah terhadap nilai ujian. *Sleep_Hours* bahkan menunjukkan korelasi negatif yang sangat kecil, menandakan bahwa durasi tidur tidak memberikan pengaruh linear yang signifikan terhadap hasil ujian siswa. Meskipun demikian, tidak terdapat pasangan fitur numerik yang memiliki korelasi sangat tinggi antara satu sama lain, sehingga risiko multikolinearitas dalam model relatif rendah dan seluruh fitur dapat digunakan tanpa perlu dihapus.

2.2 Analisis Fitur Kategorikal

2.2.1 Distribusi Fitur Kategorikal (Univariate)

Untuk menganalisis fitur-fitur kategorikal secara *univariate* dalam dataset, digunakan visualisasi berupa countplot. Countplot digunakan untuk menampilkan distribusi frekuensi setiap kategori pada masing-masing variabel kategorikal. Analisis ini membantu mengidentifikasi apakah suatu fitur memiliki distribusi kelas yang seimbang atau tidak, memahami pola kemunculan kategori tertentu, serta mendeteksi potensi ketidakseimbangan (*class imbalance*) yang dapat memengaruhi

kinerja model. Dengan memeriksa distribusi kategori melalui countplot, proses pemilihan metode encoding dan strategi preprocessing dapat dilakukan dengan lebih tepat.



(contoh visualisasi countplot)

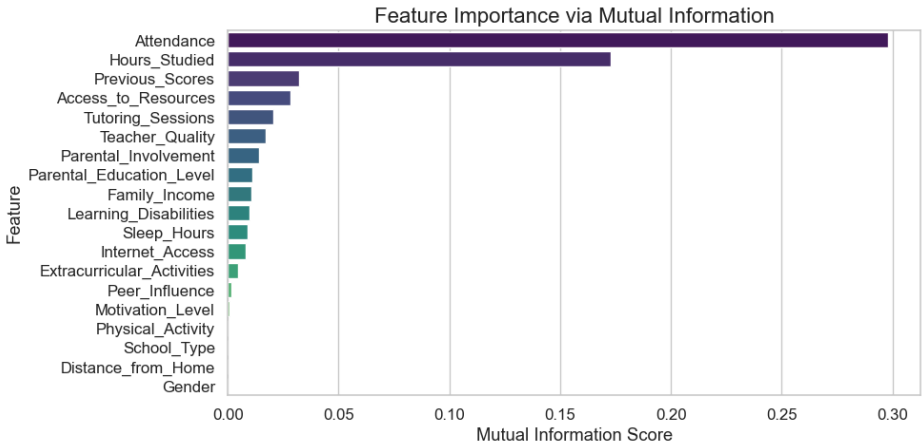
Hasil Analisis

Analisis distribusi fitur kategorikal melalui countplot menunjukkan bahwa setiap variabel memiliki pola penyebaran kategori yang berbeda. Beberapa fitur seperti *Parental_Involvement* dan *Access_to_Resources* memiliki distribusi yang relatif seimbang meskipun kategori Medium mendominasi, sedangkan fitur lain seperti *Internet_Access*, *Learning_Disabilities*, dan *School_Type* menunjukkan ketidakseimbangan yang cukup besar, di mana terdapat kategori mayoritas yang sangat dominan. Pola serupa terlihat pada *Peer_Influence*, *Teacher_Quality*, dan *Family_Income*, di mana kategori tertentu muncul jauh lebih sedikit dibandingkan kategori lain. Sementara itu, fitur seperti *Gender* memiliki distribusi yang cenderung seimbang.

Hasil ini memberikan informasi penting bagi tahap preprocessing, terutama dalam pemilihan metode encoding yang sesuai. Selain itu, countplot membantu mengidentifikasi fitur dengan variasi rendah atau kategori minor yang sangat sedikit, yang dapat mempengaruhi kekuatan sinyal fitur tersebut di dalam model dan menjadi perhatian dalam analisis potensi bias.

2.3 Analisis Mutual Information (MI)

Mutual Information (MI) digunakan untuk mengukur hubungan non-linear antara fitur input dan variabel target *Exam_Score*. Berbeda dengan korelasi Pearson yang hanya menangkap hubungan linear, MI mampu mendeteksi ketergantungan yang lebih kompleks, termasuk interaksi non-linear antara variabel kategorikal dan numerikal. Oleh karena itu, MI menjadi pelengkap penting dalam proses eksplorasi data, terutama ketika beberapa fitur tidak menunjukkan hubungan linear yang kuat



Hasil analisis menunjukkan bahwa *Attendance* (MI = 0.298) dan *Hours_Studied* (MI = 0.173) merupakan dua fitur dengan tingkat informasi tertinggi terhadap performa siswa. Fitur lainnya memiliki nilai MI yang jauh lebih rendah, menandakan bahwa kontribusinya terhadap nilai ujian sangat terbatas. Temuan ini memperkuat kesimpulan bahwa performa akademik dalam dataset ini terutama dipengaruhi oleh kehadiran dan aktivitas belajar siswa, sementara faktor-faktor kategorikal seperti motivasi, jenis sekolah, atau pendapatan keluarga tidak menunjukkan hubungan non-linear yang berarti.

BAB III
PREPROCESSING, MODELLING, dan EVALUASI

3.1 Pendahuluan Preprocessing

Tahap preprocessing dilakukan untuk memastikan bahwa seluruh fitur dalam dataset berada dalam format yang dapat digunakan oleh model machine learning. Dataset yang digunakan terdiri dari fitur numerik, kategorikal ordinal, kategorikal nominal, dan fitur biner. Setiap jenis fitur memerlukan teknik penanganan yang berbeda, sehingga proses preprocessing dirancang menggunakan *pipeline* yang terintegrasi untuk menjaga konsistensi dan mencegah *data leakage*.

3.2 Pengelompokan dan Transformasi Fitur

Sebelum melakukan transformasi, seluruh fitur diklasifikasikan ke dalam beberapa kategori berdasarkan karakteristiknya, yaitu fitur numerik, binary (2 kelas), ordinal (multiclass yang berurutan), dan nominal (multiclass tanpa urutan)

```
# 1. FEATURE GROUPS
numeric_cols = [
    'Hours_Studied', 'Attendance', 'Sleep_Hours', 'Previous_Scores',
    'Tutoring_Sessions', 'Physical_Activity'
]

binary_cols = [
    'Extracurricular_Activities', 'Internet_Access', 'School_Type',
    'Learning_Disabilities', 'Gender'
]

ordinal_cols = [
    'Parental_Involvement', 'Access_to_Resources', 'Motivation_Level',
    'Family_Income', 'Teacher_Quality',
    'Parental_Education_Level', 'Distance_from_Home'
]

nominal_cols = ['Peer_Influence']
```

Penanganan fitur binary: Menggunakan **binary mapping** (0/1).

Penanganan fitur ordinal : Menggunakan **OrdinalEncoder** dengan urutan yang ditentukan manual.

Penanganan fitur nominal : Menggunakan **OneHotEncoder(drop="first")**

3.3 Penanganan Missing Values

Dataset memiliki beberapa nilai hilang pada kolom *Teacher_Quality, Parental_Education_Level,* dan *Distance_from_Home.* Karena fitur yang memiliki missing values adalah fitur ordinal, kita mengisinya dengan nilai paling umum (modus)

```
ordinal_pipeline = Pipeline([
    ("imputer", SimpleImputer(strategy="most_frequent")),
    ("ordinal", OrdinalEncoder(categories=ordinal_categories))
])
```

3.4 Pipeline Preprocessing & Splitting Data

Seluruh tahap preprocessing digabung menggunakan ColumnTransformer agar preprocessing dilakukan langsung saat training, tidak terjadi data leakage, transformasi konsisten pada train dan test set, kode rapi dan mudah dipindahkan ke deployment. Data kemudian dibagi menjadi 80% data training dan 20% data testing

3.5 Pemilihan Algoritma Model

Setelah mencoba beberapa model, kami memilih **XGBoost** sebagai mode utama untuk projek ini. Pemilihan XGBoost sebagai model utama didasarkan pada beberapa pertimbangan. Dataset yang digunakan bersifat tabular, sehingga XGBoost lebih efektif dibandingkan neural network. Selain itu, hasil EDA menunjukkan bahwa hubungan antara fitur dan target tidak sepenuhnya linear, dengan distribusi nilai ujian yang relatif sempit, sehingga model linear berpotensi kurang mampu menangkap pola yang lebih kompleks. XGBoost juga memiliki keunggulan dalam menangani interaksi antar fitur, robust terhadap outlier, serta dilengkapi dengan regularisasi bawaan untuk mengurangi risiko overfitting. Di samping itu, XGBoost kompatibel dengan pipeline sklearn, sehingga proses preprocessing dan pemodelan dapat diintegrasikan secara efisien dalam satu alur kerja.

3.6 Hyperparameter Tuning

Untuk meningkatkan performa model, dilakukan hyperparameter tuning menggunakan RandomizedSearchCV karena lebih cepat dan efisien dibanding GridSearchCV, mampu menjelajahi ruang parameter yang luas, serta cocok untuk model kompleks seperti XGBoost

Parameter yang dituning:

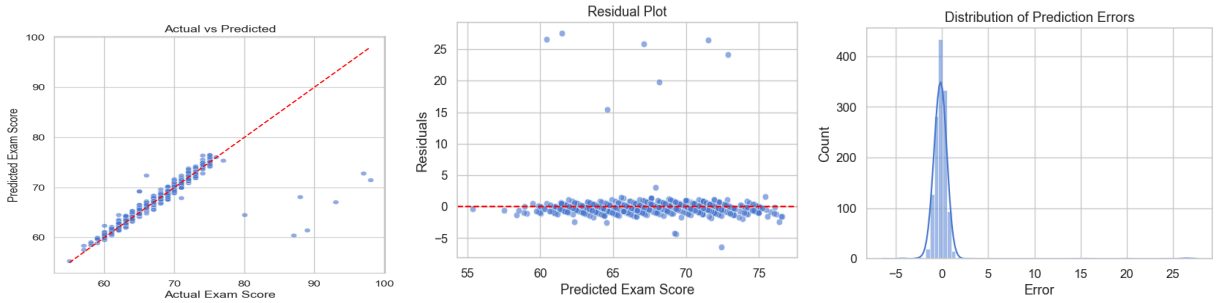
- jumlah pohon: n_estimators
- tingkat pembelajaran: learning_rate
- kedalaman pohon: max_depth
- sampling baris: subsample
- sampling kolom: colsample_bytree
- regularisasi: gamma, min_child_weight

3.8 Evaluasi Model

Setelah hyperparameter tuning menemukan konfigurasi terbaik, model terbaik menghasilkan error cross-validation: **0.6533**

Perlu ditegaskan bahwa nilai MAE pada model regresi ini dinyatakan dalam satuan yang sama dengan variabel target, yaitu poin nilai ujian, bukan dalam bentuk persentase. Oleh karena itu, nilai MAE sebesar 0.5861 menunjukkan bahwa secara rata-rata prediksi model meleset sekitar ±0.6 poin dari nilai ujian sebenarnya. Jika dibandingkan dengan skala nilai ujian yang berada pada rentang 55–100 atau terhadap nilai rata-rata ujian, besaran error ini tergolong sangat kecil. Untuk memberikan interpretasi yang lebih intuitif, error tersebut setara dengan kurang dari 1% dari rata-rata nilai ujian.

Setelah itu, dilakukan analisis error melalui tiga visualisasi utama yaitu *Actual vs Predicted Scatterplot*, *Residual Plot*, dan *Error Distribution*.



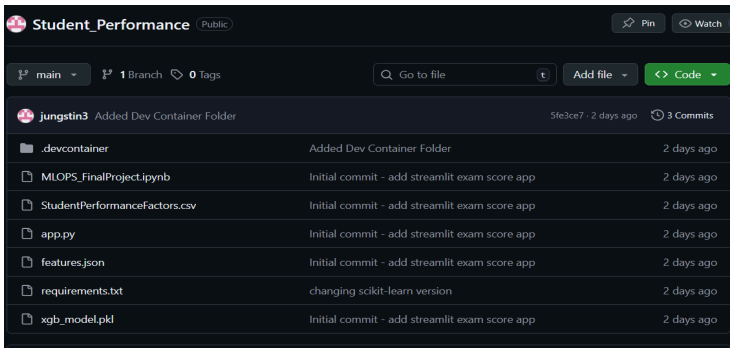
Plot *Actual vs Predicted* menunjukkan bahwa sebagian besar prediksi model berada dekat dengan garis diagonal, yang merepresentasikan prediksi sempurna. Hal ini menandakan bahwa model mampu memprediksi nilai ujian secara akurat dan konsisten di seluruh rentang nilai, tanpa adanya bias sistematis. Residual plot memperlihatkan sebaran error yang acak di sekitar garis nol dengan variansi yang relatif stabil, mengindikasikan bahwa model tidak mengalami bias prediksi maupun masalah heteroskedastisitas.

Sementara itu, histogram residual menunjukkan distribusi error yang berpusat di sekitar nol dan relatif simetris, dengan mayoritas kesalahan berada pada rentang kecil. Secara keseluruhan, ketiga visualisasi tersebut mengonfirmasi bahwa model memiliki performa yang stabil, tidak bias, dan menghasilkan kesalahan prediksi yang rendah.

BAB IV
DEPLOYMENT

Deployment merupakan tahap akhir dalam pengembangan sistem prediksi nilai ujian mahasiswa, di mana model machine learning yang telah dibangun dan dievaluasi diimplementasikan ke dalam sebuah aplikasi yang dapat digunakan secara langsung oleh pengguna. Pada penelitian ini, proses deployment dilakukan dengan mengintegrasikan model prediksi berbasis XGBoost ke dalam aplikasi web menggunakan framework Streamlit. Aplikasi tersebut kemudian di-*deploy* ke layanan Streamlit Cloud sehingga dapat diakses secara daring dan digunakan untuk melakukan prediksi nilai ujian berdasarkan data input pengguna.

4.1 Persiapan Lingkungan Deployment

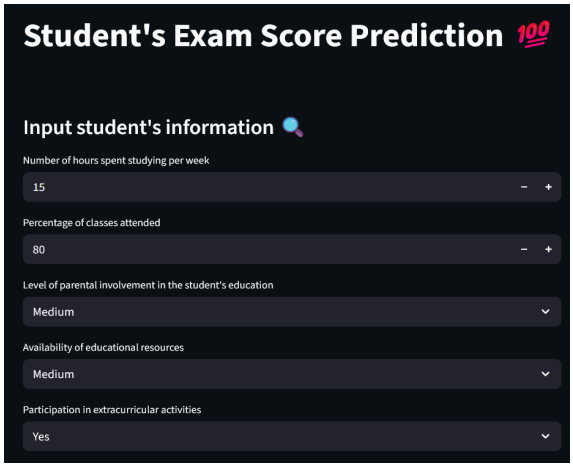


Pada tahap persiapan deployment, repository GitHub disiapkan sebagai media penyimpanan kode sumber aplikasi. Repository ini berisi file utama aplikasi Streamlit (app.py), model machine learning yang telah dilatih (xgb_model.pkl), serta file pendukung berupa daftar fitur (features.json). Selain itu, dibuat file requirements.txt yang mencantumkan seluruh library Python yang digunakan, seperti streamlit, pandas, numpy, scikit-learn, dan xgboost, guna memastikan konsistensi lingkungan antara pengembangan dan deployment.

4.2 Implementasi Aplikasi

Aplikasi dikembangkan menggunakan framework Streamlit untuk menyediakan antarmuka web yang interaktif dan mudah digunakan. Pengguna dapat memasukkan data karakteristik siswa melalui komponen input yang sesuai, seperti *number input* untuk fitur numerik dan *select box* untuk fitur kategorikal dan biner. Data yang dimasukkan kemudian diproses dan disusun dalam format yang sesuai dengan kebutuhan model. Setelah itu, aplikasi menampilkan hasil prediksi nilai ujian secara langsung kepada pengguna.

Tampilan aplikasi :



4.3 Integrasi Model dan Deployment ke Cloud

Model machine learning XGBoost yang telah dilatih disimpan dalam format .pkl dan dimuat ke dalam aplikasi Streamlit saat aplikasi dijalankan. File features.json digunakan untuk memastikan kesesuaian struktur dan urutan fitur antara data input pengguna dan model. Data input yang diterima dari antarmuka aplikasi dikonversi ke dalam bentuk DataFrame, disesuaikan dengan fitur model, dan kemudian digunakan untuk menghasilkan prediksi nilai ujian.

Aplikasi selanjutnya dideploy menggunakan Streamlit Cloud sebagai platform cloud deployment. Platform ini dipilih karena kemudahan konfigurasi dan dukungan langsung terhadap aplikasi Streamlit, sehingga proses deployment dapat dilakukan secara cepat tanpa memerlukan pengaturan server yang kompleks. Dengan deployment ini, aplikasi dapat diakses secara online dan digunakan untuk melakukan prediksi nilai ujian secara real time.

Setelah proses build dan deployment selesai, aplikasi dapat diakses melalui URL publik yang disediakan oleh Streamlit Cloud. URL ini memungkinkan pengguna untuk mengakses aplikasi prediksi nilai ujian secara daring tanpa perlu melakukan instalasi tambahan.

Selain itu, Streamlit Cloud mendukung fitur *auto redeploy*, di mana setiap perubahan pada kode aplikasi di repository GitHub akan secara otomatis memperbarui aplikasi yang telah di-deploy. Fitur ini mendukung praktik MLOps, khususnya *continuous integration* dan *continuous deployment* (CI/CD), sehingga aplikasi selalu menggunakan versi kode dan model terbaru tanpa memerlukan proses deployment manual secara berulang.

4.4 Pengujian Hasil Deployment

Setelah aplikasi berhasil di-deploy ke Streamlit Cloud, dilakukan pengujian untuk memastikan bahwa seluruh komponen sistem berjalan dengan baik dan sesuai dengan tujuan pengembangan. Pengujian dilakukan dengan memasukkan berbagai kombinasi data input yang merepresentasikan kondisi siswa yang berbeda, guna mengevaluasi fungsionalitas, akurasi prediksi, serta stabilitas aplikasi.

Kami membuat 3 data dummy, yang terdiri dari 1 siswa dengan performa buruk, 1 performa biasa saja, dan 1 performa luar biasa, dan hasilnya cukup akurat, yaitu 55 untuk siswa pertama, 65 untuk siswa kedua (mendekati mean), dan 91 untuk siswa ketiga.

BAB V
PENUTUP

5.1 Kesimpulan

Pada proyek ini telah berhasil dikembangkan sebuah model machine learning untuk memprediksi nilai ujian siswa berdasarkan berbagai faktor akademik dan non-akademik. Proses pengembangan dimulai dari Exploratory Data Analysis (EDA) untuk memahami karakteristik data, dilanjutkan dengan preprocessing yang mencakup penanganan missing values, pemilihan metode encoding yang sesuai, serta normalisasi fitur numerik. Model XGBoost dipilih sebagai model utama karena kemampuannya dalam menangani data tabular dan hubungan non-linear antar fitur. Hasil evaluasi menunjukkan bahwa model memiliki performa yang baik dan stabil, dengan nilai MAE yang rendah serta kemampuan menjelaskan sebagian besar variasi performa siswa. Model juga berhasil diintegrasikan ke dalam aplikasi web berbasis Streamlit dan dideploy ke cloud sehingga dapat digunakan secara real time.

5.2 Saran

Untuk pengembangan selanjutnya, kualitas dan variasi dataset dapat ditingkatkan agar model mampu menangkap pola yang lebih kompleks, khususnya dengan menambahkan fitur yang lebih relevan atau data lintas waktu. Selain itu, eksplorasi feature selection dan feature engineering lanjutan dapat dilakukan untuk mengurangi noise dan meningkatkan performa model. Dari sisi MLOps, pengembangan dapat diperluas dengan menambahkan pemantauan performa model setelah deployment serta mekanisme pembaruan model secara berkala.

LAMPIRAN

Link aplikasi: https://studentperformancepred1ctor.streamlit.app/

Link Repository Github: https://github.com/jungstin3/Student_Performance

Link video demo aplikasi:
https://drive.google.com/file/d/14uti54WfomDUvkvIjGnwP54RHPM_WReM/view?usp=sharing

APPENDIX

Tabel Kontribusi Tim

Brandon Tumiwa	**Justine Ria Jingga**	**Stanley Angkasa**	**Raka Priyahita Pramudito**
Membahas ide projek	Membahas ide projek	Membahas ide projek	Membahas ide projek
Melakukan eksplorasi dataset (EDA)	Mencari dataset untuk projek	Menulis bab pendahuluan laporan	Mencari dataset untuk projek
Melakukan modelling dan evaluasi model	Melakukan eksplorasi dataset (EDA)	Melakukan dan menulis bagian eksplorasi dataset (EDA)	Menulis bab pendahuluan laporan
Menulis bab 2 laporan (EDA)	Membuat github repo dan melakukan *deployment,* serta membuat video demo	Menulis bab kesimpulan dan saran laporan	Menulis bab 2 laporan (deskripsi dataset)
Menulis bab 3 laporan (preprocessing,model, dan evaluasi)	Menulis bab 4 laporan (*deployment*)	Membuat *Power Point* untuk presentasi	Membuat *Power Point* untuk presentasi
Membuat *Power Point* untuk presentasi	Membuat *Power Point* untuk presentasi	Mendiskusikan hasil projek dan melakukan revisi	Mendiskusikan hasil projek dan melakukan revisi
Mendiskusikan hasil projek dan melakukan revisi	Mendiskusikan hasil projek dan melakukan revisi		