NAAC Grade 'A"

R. C. PATEL
INSTITUTE OF TECHNOLOGY

An Autonomous Institute

MANTHAN

Department of Computer Science & Engineering (Data Science)

ISSUE YEAR
2024-25

"To understand God's thoughts we must study statistics, for these are the measure of His purpose."

Florence Nightingale
Data Visualization Pioneer

Vision-Mission

Institute Vision

To become a leading Institute in Technical education fostering innovation, research, ethical values, and sustainable development for the betterment of society.

Institute Mission

To impart high quality Technical Education through :

- Innovative and Interactive learning process and high quality, internationally recognized instructional programs.
- Fostering a scientific temper among students by the means of a liaison with the Academia, Industries and Government.
- Preparing students from diverse backgrounds to have aptitude for research and spirit of Professionalism.
- To contribute to the nation’s sustainable development.

Vision-Mission

Department Vision

To provide cutting-edge Computer Engineering education in Data Science while instilling socio-moral values.

Department Mission

M1: To deliver state-of-the-art, ICT-enabled teaching and learning to achieve excellence in Data Science education.

M2: To develop professionally competent Data Science Engineers, meeting evolving industrial and societal needs.

M3: To prepare employable professionals with ethical values and a commitment to professional and social responsibility.

Visionary Insights

The Director

Prof. Dr. Jayantrao Bhaurao Patil

At RCPIT, we blend academic rigour with real-world exposure to create future-ready professionals. Emphasizing research, interdisciplinary learning, and soft skills, our hands-on approach empowers students to become ethical leaders, innovators, and entrepreneurs prepared to succeed in a rapidly evolving global landscape.

Visionary Insights

The Head of Department

Prof. Dr. Ujwala Patil

Established in 2020 at RCPIT, Shirpur, the Computer Science & Engineering (Data Science) program trains students in algorithmic thinking and statistical modeling. This interdisciplinary field blends computer science, mathematics, and business analytics. Driven by massive industry demand, it is widely recognized as a highly dynamic and lucrative global career path.

Editor's Message

Prof. Manisha S. Patil

Editor (Faculty)

DEAR READERS,

Welcome to the latest edition of our Department Magazine. This magazine is a platform to share knowledge, ideas, and achievements in Computer Science & Engineering (Data Science). It brings together expert articles, faculty contributions, career guidance from the Training and Placement Cell, and abstracts of students' publications. We hope these pages inspire curiosity, encourage innovation, and motivate every student to learn continuously. May this magazine strengthen the bond between academia and industry while preparing our students for successful and rewarding careers.

Editorial Board

Mr. Sarthak Patil

Ms. Riya Deshmukh

Mr. Tejas Chaudhari

Contents

Technical Article

LLMs as Data Tools: Opportunities and Hallucination Risks

Prof. P. D. Lanjewar

Large Language Models have changed what data scientists can accomplish daily. Tasks that once required custom pipelines and hours of specialised work can now be done with a well-written prompt. This is a genuine productivity revolution. But it also introduces new risks that every data professional must understand clearly.

This article explores how data scientists use LLMs as analytical tools, where they add real value, and how to guard against the most serious risk: hallucination.

What LLMs Can Do for Data Scientists

LLMs are excellent at tasks that involve language and structure. They can convert a plain-English question into a SQL query. A business analyst without SQL skills can type a request and receive a ready-to-run query. This makes data access possible for everyone in an organisation, not just technical staff.

LLMs can also summarise large collections of text. A data scientist analysing thousands of customer support tickets can ask an LLM to extract common complaint themes and flag the most critical cases. A task that once took a week can now take an hour.

LLMs can generate Python code for data wrangling, write documentation, and explain complex model outputs in plain language. These capabilities genuinely expand what a single data scientist can deliver in a working day.

Fig. 1— LLM-augmented analytics pipeline with a hallucination check and retry loop

The Hallucination Problem

LLMs generate text by predicting the most probable next token. They do not retrieve facts from a verified database. They produce plausible-sounding text based on learned patterns.

When a model does not know something, it often continues generating anyway. It produces fluent, confident, but entirely wrong content. This is called hallucination. In a data context, a hallucinating LLM might generate SQL with a table or column name that does not exist.

Technical Article

LLMs as Data Tools: Opportunities and Hallucination Risks

It might state a statistical fact that sounds credible but is fabricated. It might summarise a report with invented details not present in the source. These errors are dangerous because they look exactly like correct output.

A junior analyst who trusts an LLM-generated SQL query without checking it may insert wrong numbers into an executive dashboard. The consequences can range from embarrassing to costly.

Strategy 1: Grounding

The most effective defence against hallucination is grounding. Provide the LLM with the actual data it needs directly within the prompt. Instead of asking a general question, paste the relevant rows or schema into the prompt.

When the model has the real data in front of it, it cannot hallucinate figures it needs. This is the core idea behind Retrieval-Augmented Generation (RAG). The model is given retrieved context before it answers. It synthesises, rather than fabricates.

For SQL generation, always include the full table schema in the prompt. Specify column names, data types, and any relevant constraints. The more context you provide, the fewer assumptions the model must make on its own.

Strategy 2: Output Validation

Never use LLM-generated SQL without testing it first. Run the query and check whether the result makes sense against known benchmarks. A query returning zero rows when the dataset has thousands of entries is a clear warning sign.

For factual claims, cross-reference against authoritative sources before use. Treat every LLM output as a first draft that requires verification, not a finished product ready for presentation.

Build a validation step into your pipeline. After the LLM generates an output, a secondary check compares it against simple sanity rules. Does the query reference only tables that exist? Do the numerical outputs fall within expected ranges? Automated validation catches the most common hallucination patterns quickly.

Strategy 3: Using the Right Tool

LLMs are not universal tools. For tasks with a single unambiguous correct answer, such as sorting a list or computing a median, use deterministic code.

Technical Article

LLMs as Data Tools: Opportunities and Hallucination Risks

Python gives you exact answers without any risk of hallucination.

For tasks involving language understanding, flexible text extraction, or summarisation of unstructured documents, LLMs genuinely add value. The skilled data scientist in 2024 knows which category each task belongs to.

A helpful mental model: use LLMs for language tasks and Python for computation tasks. When a task requires both, use Python to prepare the data and structure the problem, then pass the well-defined language task to the LLM.

Prompt Engineering for Reliable Outputs

The quality of an LLM output depends heavily on prompt quality. Vague prompts produce vague outputs. Specific, structured prompts with explicit instructions and examples produce far more reliable results.

Ask the model to respond in a structured format such as JSON. Specify exactly which fields you need. Instruct the model to say "I do not know" rather than guessing when information is missing. These simple instructions dramatically reduce hallucination rates.

Document your best-performing prompts as reusable templates. Test them on a set of known inputs before trusting them with new data. Treat prompt development with the same rigour you would apply to any other piece of production code.

The Professional Discipline That Does Not Change

As LLM capabilities grow, their role in the data science workflow will only expand. But the core discipline of verifying outputs, understanding limitations, and applying critical judgment to every result remains constant.

The best data scientists of 2024 combine genuine LLM fluency with the rigour that has always defined good analytical work. Curiosity and scepticism must coexist. Use these tools boldly, and verify what they produce carefully.

Technical Article

Algorithmic Fairness: When Data Reflects Bias

Prof. P. S. Sanjekar

Machine learning models do not operate in a vacuum. They learn from data generated by human societies. Those societies have histories of inequality and structural bias. When models learn from this data, they can inherit and amplify the very injustices they were never designed to perpetuate.

Understanding algorithmic fairness is no longer optional for data scientists. It is a professional and ethical responsibility. This article introduces the core concepts, shows where bias enters the pipeline, and outlines practical strategies for building fairer systems.

How Bias Enters the Data

Bias enters datasets in several distinct ways. Historical bias occurs when past human decisions reflected discrimination. A hiring dataset recording who was hired in the 1990s encodes the prejudices of those recruiters. Training a model on this data reproduces those biases in future hiring recommendations.

Representation bias occurs when certain groups are underrepresented in the training data. A facial recognition system trained mostly on lighter-skinned faces performs poorly on darker-skinned faces. This is not a deliberate design choice. It is the consequence of who was included or excluded during data collection.

Measurement bias occurs when the proxy variable used as a label is itself biased. Using arrest records as a proxy for criminal behaviour introduces police patrol patterns as a confounding factor. Neighbourhoods with heavier policing accumulate more arrest records regardless of underlying crime rates.

Fairness Definitions

Demographic parity requires the model to produce positive predictions at equal rates across all demographic groups. Equalised odds requires both the true positive rate and the false positive rate to be equal across groups. Individual fairness requires that similar individuals receive similar predictions, regardless of group membership.

These definitions sound straightforward but are mathematically incompatible when base rates differ between groups. Choosing the right fairness criterion requires understanding the context and the real-world consequences of each type of error.

Technical Article

Algorithmic Fairness: When Data Reflects Bias

Involving affected communities in this choice is essential. Data scientists cannot make this decision alone. The people affected by a system have a legitimate voice in defining what fairness means for their situation.

Practical Bias Auditing

Before deploying any model that makes consequential decisions about people, conduct a bias audit. Split your test set by demographic group. Compute model performance separately for each group. Are accuracy, recall, and precision comparable across all groups?

Large disparities signal potential unfairness and demand investigation. Tools like Fairlearn and IBM AI Fairness 360 provide standard fairness metrics, visualisations of group disparities, and automated mitigation techniques. Make bias auditing a standard step in your model evaluation process, not an afterthought added under pressure.

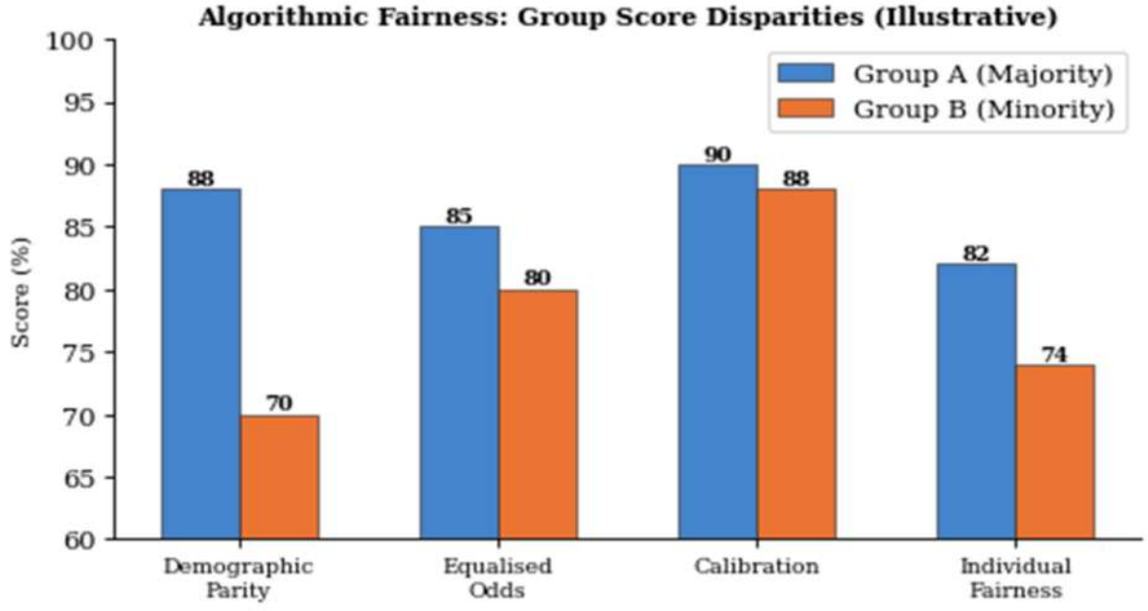

Fig. 2 — Illustrative group score disparities across four common fairness metrics

Mitigation Strategies

Pre-processing techniques adjust the training data before model training. Re-weighting assigns higher importance to underrepresented groups during training.

Technical Article

Algorithmic Fairness: When Data Reflects Bias

Re-sampling balances group representation in the dataset. These approaches are simple and model-agnostic but do not guarantee fairness after training completes.

In-processing techniques modify the training objective itself. Adding a fairness penalty to the loss function forces the model to treat disparate outcomes as a cost to minimise, alongside prediction error. This directly encodes fairness as a training goal.

Post-processing techniques adjust model thresholds separately for different demographic groups to equalise a chosen fairness metric after training. This approach is flexible and does not require retraining but may reduce overall accuracy to achieve fairness.

A Real-World Ex ample

Consider a loan approval model trained on historical bank data. The historical data reflects decades of discriminatory lending practices. A model trained naively on this data denies loans to minority applicants at a higher rate than equally creditworthy majority applicants.

A bias audit reveals the disparity. The team applies re-weighting to increase the influence of minority applicants during training. After retraining, approval rates become more equitable. A post-deployment audit six months later confirms the improvement has held in practice. This iterative process of audit, mitigation, and re-audit is the real workflow of responsible AI development.

Beyond the Metrics

Fairness is not purely a technical problem. No mathematical metric can capture the full complexity of social justice. Technical interventions must be accompanied by diverse development teams, inclusive design processes, and meaningful engagement with affected communities.

Data scientists have real power to shape systems that affect employment, housing, credit, healthcare, and justice. That power comes with responsibility. Learning the technical vocabulary of fairness is the necessary first step. Applying it with humility and in collaboration with those affected is the ongoing commitment.

Bias in AI is not inevitable. It is a consequence of choices made during data collection, feature selection, model training, and deployment. Recognising where those choices occur is the first step toward making better ones.

Technical Article

Time Series Forecasting for Engineering Applications

Prof. A. K. Patil

Time series data appears throughout engineering. Sensor readings from industrial machines, electricity consumption logs, environmental monitoring outputs, and traffic flow measurements are all time series. Forecasting these sequences accurately supports better maintenance decisions, resource allocation, and system control.

This article introduces two main approaches to time series forecasting: the classical ARIMA model and the modern LSTM neural network. You will understand when to use each and how to implement both in Python.

What Makes Time Series Special?

Time series data violates a core assumption of most machine learning models: that observations are independent of one another. In a time series, each observation is correlated with previous observations. Tomorrow's temperature depends on today's temperature. Next month's electricity demand depends on the patterns of previous months.

A time series also carries internal structure that standard ML models cannot capture without explicit help. Trend is the long-term direction of the series. Seasonality is the repeating pattern that follows a fixed cycle, such as daily or yearly rhythms. Noise is the random variation around the underlying signal.

ARIMA: The Classical Approach

ARIMA stands for AutoRegressive Integrated Moving Average. It has three components. The AR term models the relationship between the current value and its previous values. The I term handles differencing to make the series stationary. The MA term models the relationship between the current value and past forecast errors.

ARIMA is interpretable, well-understood, and reliable for univariate time series with broadly linear structure. It is fast to train and requires minimal data. Use ARIMA as your starting baseline before experimenting with more complex models.

Selecting the correct ARIMA parameters (p, d, q) is the main challenge. Plot the autocorrelation function (ACF) and partial autocorrelation function (PACF) to guide your parameter choices. The Python library statsmodels provides auto_arima() which searches for the best parameters automatically.

Technical Article

Time Series Forecasting for Engineering Applications

LSTM: Learning Sequential Patterns

Long Short-Term Memory networks are a type of recurrent neural network designed to learn long-term dependencies in sequences. They maintain a cell state that carries information across many time steps. Three learned gates control what information is stored, discarded, or output at each step: the input gate, the forget gate, and the output gate.

LSTMs excel when the time series has complex non-linear patterns, when you have multiple input features (multivariate forecasting), and when you have enough data to train the network reliably. For a single sensor with only a few hundred data points, ARIMA often outperforms an LSTM because the LSTM needs more data to generalise well.

Implementing LSTM Forecasting

Prepare your data by creating sliding windows. To predict the next value using the previous 24 time steps, build samples where each input is a sequence of 24 consecutive readings and the label is the 25th. Normalise all values to a common range using MinMaxScaler before training. This prevents larger-valued features from dominating smaller ones.

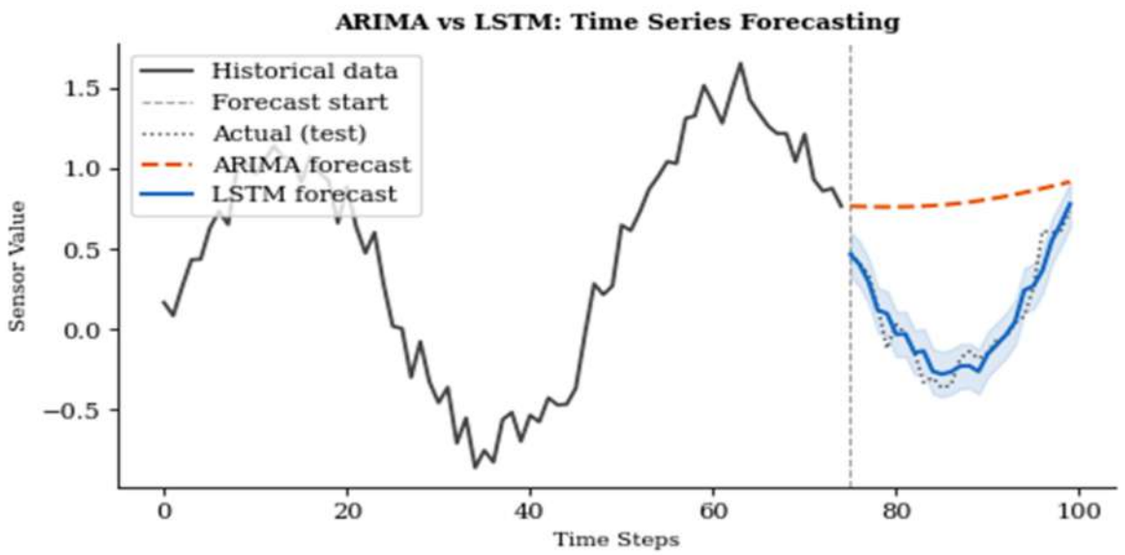

Fig. 3 — ARIMA versus LSTM forecasting on a synthetic sensor time series

In Keras, build a simple LSTM: one LSTM layer with 64 units, a Dropout layer at 0.2 rate, and a Dense output layer with one neuron. Compile with Adam optimiser and Mean Squared Error loss.

Technical Article

Time Series Forecasting for Engineering Applications

Train for 50 epochs with a 20% validation split. Monitor validation loss to detect overfitting early.

Engineering Applications

Predictive maintenance uses LSTM models to forecast vibration or temperature anomalies in rotating machinery. The model learns the normal operating signature of a machine. When a forecast diverges significantly from actual readings, a maintenance alert fires before the actual breakdown occurs.

Energy management systems use time series models to forecast electricity demand for the next 24 hours. This forecast drives generator scheduling. An accurate 24-hour demand forecast can reduce fuel costs by 5 to 15 percent in thermal power plants by avoiding unnecessary generator start-ups and shutdowns.

Environmental monitoring stations forecast air quality indices. When a model predicts that PM2.5 will exceed safe limits in the next six hours, authorities can issue public health advisories proactively. The model gives decision-makers time to act before the situation peaks.

Choosing Between ARIMA and LSTM

Use ARIMA when your series is univariate, relatively short (a few hundred observations), and broadly linear in structure. ARIMA is also the better choice when interpretability matters, since its parameters have clear mathematical meanings.

Use LSTM when your series is long (thousands of observations or more), involves multiple input features, or contains complex non-linear patterns that ARIMA cannot capture. LSTM is also preferable when you want a single model architecture that handles many different sensors simultaneously.

In practice, always try both. Compute RMSE and MAE on a held-out test set for each approach. Choose the model that performs better on the specific series you are forecasting. There is no universal winner between these two approaches.

Technical Article

Explainable AI: Making Black-Box Models Interpretable

Prof. M. S. Patil

Modern machine learning models can be extraordinarily accurate. Deep neural networks and gradient-boosted trees routinely outperform simpler models on complex tasks. But there is a serious problem. The more powerful the model, the harder it is to understand why it makes any particular prediction.

This opacity is called the black-box problem. In high-stakes domains such as healthcare, finance, and criminal justice, an unexplainable model decision is often legally and ethically unacceptable. Explainable AI, or XAI, is the field dedicated to providing those explanations in human-understandable terms.

Why Interpretability Matters

Interpretability matters for three concrete reasons. First, it builds trust. A doctor who can see which symptoms most influenced a diagnosis recommendation will trust the model enough to act on it. A doctor facing an opaque output will rightly refuse.

Second, it enables debugging. When a model makes a surprising error, interpretability tools reveal which features drove the wrong prediction. That understanding is essential for improvement. A feature with unexpectedly high influence often points to a data quality issue or a spurious correlation in the training data.

Third, it is increasingly required by regulation. The European Union GDPR includes a right to explanation for automated decisions affecting individuals. India's data protection legislation is moving in a similar direction. Data scientists who cannot explain their models face growing regulatory barriers to deployment.

LIME: Local Interpretable Model-Agnostic Ex planations

LIME explains individual predictions by building a simple, interpretable model that approximates the complex model's behaviour locally around the specific data point of interest. The key insight is that even a black-box model behaves approximately linearly in the immediate neighbourhood of any single input.

To explain one prediction, LIME perturbs the input slightly in many ways, collects the complex model's predictions on these perturbed inputs, and fits a weighted linear regression to approximate which features drove that specific prediction.

Technical Article

Explainable AI: Making Black-Box Models Interpretable

The result is a ranked list of features and their positive or negative contributions to that individual output.

LIME is model-agnostic: it works with any classifier or regressor regardless of internal architecture. Install it with pip install lime. It requires no changes to your existing model or training process.

SHAP: SHapley Additive ex Planations

SHAP values are grounded in the concept of Shapley values from cooperative game theory. They provide a mathematically principled attribution of each feature's contribution to a model's prediction. Every feature receives a SHAP value measuring how much it pushed the prediction above or below the average prediction across all data points.

SHAP has several important advantages over LIME. It is consistent: if a model changes so that a feature becomes more influential, its SHAP value always increases. It is locally accurate: SHAP values sum exactly to the difference between the individual prediction and the global baseline. These mathematical guarantees make SHAP the preferred tool for serious explainability work.

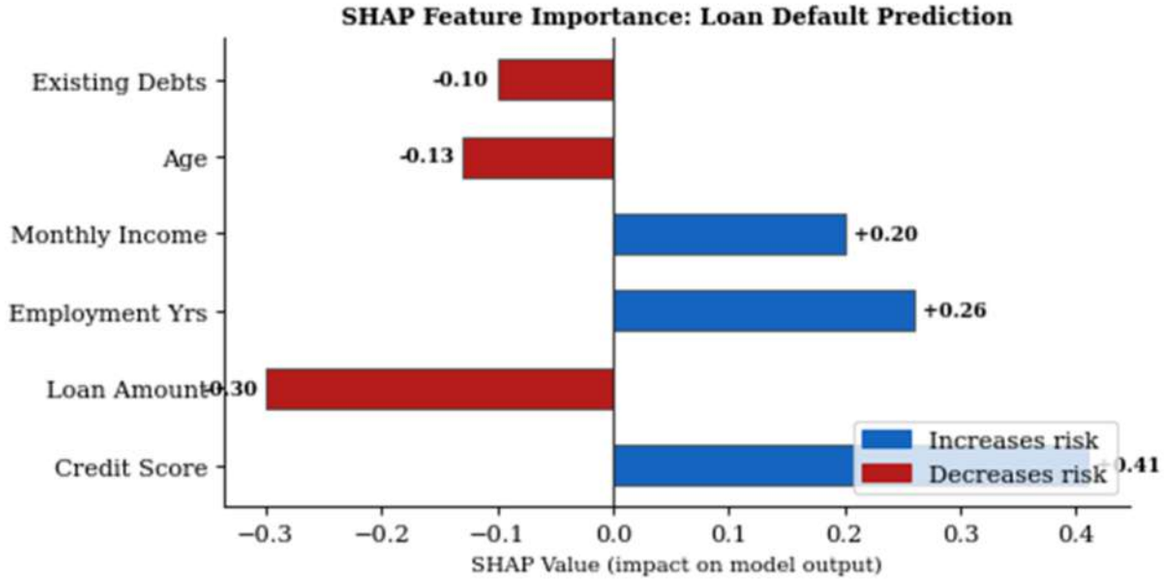

Fig. 4 — SHAP values for a loan default model: blue increases predicted risk, red decreases it

Technical Article

Explainable AI: Making Black-Box Models Interpretable

Practical SHAP Implementation

Install SHAP with pip install shap. Import the library and create an explainer matched to your model type. For tree-based models such as Random Forest or XGBoost, use shap.TreeExplainer. This is highly optimised and runs very fast even on large datasets.

Call explainer.shap_values(X_test) to compute SHAP values for your entire test set. Each row of the resulting matrix contains one SHAP value per feature, for one test sample. Positive values mean that feature pushed the prediction upward. Negative values mean it pushed the prediction downward.

Call shap.summary_plot(shap_values, X_test) to generate a beeswarm chart. Each dot represents one data point. The horizontal position shows the SHAP value. The colour shows the original feature value (red for high, blue for low). This single plot reveals which features matter most globally and how they influence predictions across the full dataset.

Global vs Local Ex planations

A global explanation describes how the model behaves across all predictions. SHAP feature importance plots and mean absolute SHAP values provide global explanations. They answer the question: which features does this model rely on most overall?

A local explanation describes why the model made one specific prediction for one specific individual. A force plot generated by shap.force_plot() shows which features pushed a single prediction above or below the baseline. Local explanations are essential when a model decision is challenged by an individual, such as a loan rejection.

Good explainability practice provides both. Use global explanations to validate that the model is relying on sensible features. Use local explanations to communicate individual decisions to the people affected by them.

Integrating XAI into Your Workflow

Include an explainability section in every model development report you write. Document the top five most important features and the direction of their influence. Note any features with unexpectedly high SHAP values; these are worth investigating for data quality issues.

This creates an audit trail that supports review and regulatory compliance.

Explainability is not a separate step added at the end. A model that you cannot explain is a model you do not yet fully understand. Build that understanding before you deploy.

Career Horizon

Data Science in 2024: New Roles, New Skills

Prof. T. R. Girase

The data science job market in 2024 looks noticeably different from two years ago. The widespread availability of large language models has shifted the skills employers value, the roles they hire for, and the standards they apply during interviews. This article maps those changes and gives you a clear plan for staying well ahead of the curve.

The core foundations of statistics, Python programming, and data manipulation have not become less important. If anything, they are more valuable now, precisely because they remain rare when combined with LLM fluency. What has changed is the layer above these foundations.

The AI-Augmented Analyst

The most in-demand entry-level profile in 2024 is the AI-augmented analyst. This person combines traditional analytical skills with LLM tools to dramatically increase their output. They write Python code assisted by tools like GitHub Copilot. They query databases using LLM-generated SQL. They summarise research and draft briefings using LLM APIs.

The productivity difference between an analyst who uses these tools effectively and one who does not can reach two to five times. Companies that have measured this difference are now actively seeking candidates who demonstrate genuine comfort with AI-assisted workflows from day one on the job.

MLOps Engineer

As more companies move from experimental ML to production ML, demand for MLOps engineers has grown sharply. MLOps stands for Machine Learning Operations. It is the practice of deploying, monitoring, and maintaining ML models in production systems at scale.

MLOps engineers design pipelines that automate the journey from raw data to a deployed and monitored model. They use Apache Airflow for workflow orchestration, MLflow for experiment tracking and model versioning, and Kubernetes for container management. They set up monitoring dashboards that alert teams when model performance degrades.

This role is especially well-suited to students who enjoy software engineering as much as machine learning itself. Salaries for MLOps engineers with even one year of experience are among the highest in the data profession in India.

Career Horizon

Data Science in 2024: New Roles, New Skills

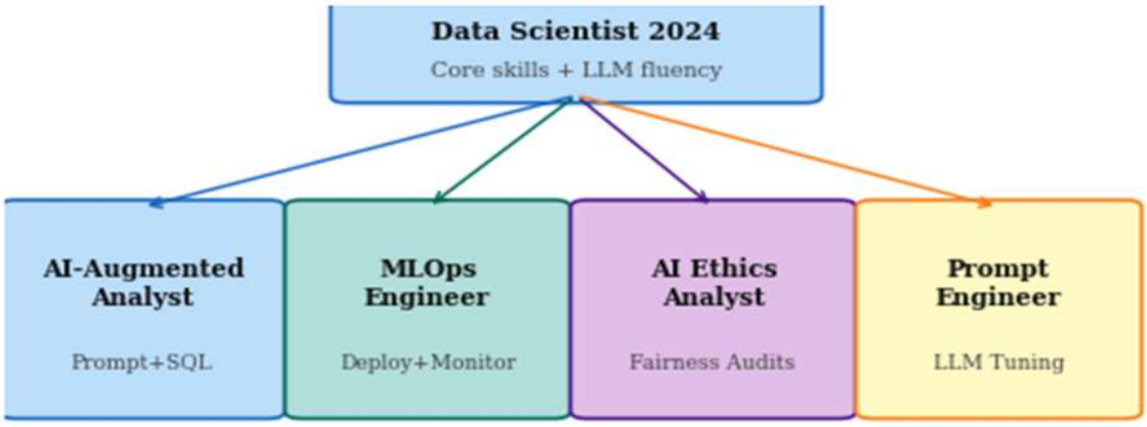

Fig. 5 — Emerging data science roles for 2024, branching from a core of LLM-augmented skills

Prompt Engineer

Prompt engineering is the practice of crafting inputs to LLMs that reliably produce desired outputs. Doing it well requires a deep understanding of how these models respond to different phrasings, structures, and examples. The same question asked in two different ways can produce results of very different quality.

Systematic prompt design includes chain-of-thought prompting, few-shot examples, and structured output formats. These techniques can dramatically improve the reliability and quality of LLM outputs. Companies building LLM-powered products actively hire people who can design, test, and iterate on prompts with engineering discipline.

AI Ethics Analyst

As AI systems make more consequential decisions, the demand for professionals who can audit them for fairness, bias, and safety has grown significantly. The AI ethics analyst designs and conducts fairness evaluations, documents model behaviour across demographic groups, and communicates findings to technical and non-technical stakeholders alike.

This role is a strong fit for data science students who are drawn to both quantitative work and questions of social impact. It requires statistical skills for bias measurement combined with communication skills for translating findings into actionable recommendations for product and engineering teams.

Career Horizon

Data Science in 2024: New Roles, New Skills

Skills That Have Not Changed

Amid all this change, certain foundations remain constant and are more valuable than ever. Statistical rigour is one. The ability to design a proper experiment, choose the right statistical test, and interpret results without over-claiming is a skill that AI tools cannot replicate reliably.

SQL remains non-negotiable. Every data company tests SQL in interviews. Students who neglect SQL and focus only on Python and ML consistently underperform in hiring pipelines. Practise SQL daily on HackerRank or LeetCode. The investment pays back quickly and consistently.

Communication is the other irreplaceable skill. Translating technical findings into clear business implications, telling a coherent story with data, and pushing back honestly when pressure conflicts with statistical reality separate career-defining data scientists from interchangeable ones.

Your 2024 Skill-Building Plan

If you are a second- or third-year student reading this in 2024, here is a concrete priority list. First, master Python, SQL, and Scikit-learn to the point where you can complete any standard data analysis or ML project independently without assistance.

Second, build working familiarity with at least one LLM API. Create three projects that incorporate LLM capabilities within a larger Python application. These could be an LLM-powered document Q&A system, an automated report summariser, or a natural language to SQL interface for a database you build yourself.

Third, learn the basics of MLOps. Deploy one project end-to-end with a simple REST API and a monitoring log. Fourth, contribute to one open-source project, even if only to fix documentation. Fifth, build your professional network by engaging genuinely in online data science communities and local meetups.

The Path Forward

The skills and habits you build in the next 18 months will shape the first decade of your career. The data science field rewards those who combine strong technical foundations with genuine curiosity and consistent, honest practice.

Selected Student Articles

Smart Contract-Based Decentralized Rental System for Housing

Abstract
Blockchain is the kind of innovation that was presented to the world a long time ago with the offer assistance of a few cryptocurrencies, but as in the assist investigate done by individuals, world realized blockchain can be utilized in for diverse purposes to make framework more secure with the offer assistance of legitimate computer program and innovation we can make anything much secure and decentralized same can be utilized in house rental framework.

Keywords – Internet, Technology, Safety, Blockchain, Smart contract, Decentralized, Real Estate.

Vedant Ugale, Hrushikesh Shinde, Vivek Bhoi, Ashutosh Pawar, and Prof. Manisha Patil

Selected Student Articles

Blockchain Based Fake Product Identification

Abstract

Counterfeit products are a growing problem across many industries, damaging company reputations, reducing profits, and even putting consumers' health at risk. Traditional methods to check if a product is real or fake often aren't reliable. In this project, we introduce a system that uses blockchain technology to help identify fake products. In our system, each product is given a unique digital code, and every step in the product's journey from manufacturing to delivery is recorded on the blockchain. This allows customers and businesses to easily check where the product has been and confirm if it's genuine. Our goal is to make it easier to track products, reduce fake goods in the market, and build more trust between companies and consumers using a secure and efficient system. Fake products have become a serious issue in today's global market, leading to financial losses for companies and putting customers at risk. Many existing methods for checking if a product is real or fake can be tricked or are hard to manage. This project presents a smart and secure solution using blockchain technology a system known for keeping data safe and unchangeable. Each product is tagged with a special digital ID, and every time it moves through the supply chain (from factory to customer), its details are saved on the blockchain. This creates a transparent history that anyone can check to confirm if the product is original. Our system helps customers trust what they're buying and gives companies a better way to protect their brand from counterfeits. It's a modern approach to fighting fake products using trusted digital tools.

Keywords – Internet, Technology, Safety, Blockchain, Smart contract, Decentralized, Real Estate.

Rohan Shirsath, Uday Shinde, Vijay Patil, and Dr. Priti S. Sanjekar