David Galati University of Twente May 22, 2024

AAIB Assignment 3

Acknowledgement: ChatGPT was used to aid the coding and answering the questions in this assignment.

Task 1

• What are the significant impacts of issues like algorithmic bias and privacy concerns in business? (Min 250 words, Max 1 A4 page)

Algorithmic bias and privacy concerns are critical issues that significantly impact businesses across various sectors. As companies increasingly rely on data-driven decisions and automated systems, addressing these challenges is crucial for maintaining ethical standards, consumer trust, and regulatory compliance.

Algorithmic bias occurs when AI systems produce results that are systematically prejudiced due to biased data or flawed algorithms. This bias can have several significant impacts on businesses. Firstly, it can cause severe reputation damage. For instance, biased algorithms can lead to discriminatory practices, such as unfair hiring processes or biased loan approvals, resulting in a loss of customer trust and negative publicity. Additionally, there are increasing legal and regulatory risks associated with algorithmic bias. Companies using biased algorithms may face legal actions and fines under laws like the EU's General Data Protection Regulation (GDPR) or the California Consumer Privacy Act (CCPA). Ensuring algorithms are fair and unbiased is crucial to avoid these legal pitfalls. Furthermore, bias in algorithms can lead to operational inefficiency by causing suboptimal business decisions, which affect overall efficiency and effectiveness. For example, biased customer segmentation can result in inappropriate marketing strategies, leading to poor customer engagement and loss of revenue. Lastly, businesses have a social responsibility to promote inclusivity and equality. Algorithmic bias undermines these principles, potentially leading to societal backlash and harming the broader community.

Privacy concerns arise from the collection, storage, and use of personal data. The impacts of mishandling privacy are equally significant. One major consequence is the erosion of consumer trust. Data breaches and privacy violations can lead to a loss of business and a damaged brand reputation, as customers expect their personal information to be safeguarded. Moreover, there are stringent regulatory compliance requirements imposed by privacy laws like GDPR and CCPA. Non-compliance can result in hefty fines and legal challenges, making it crucial for businesses to implement robust data protection measures. Beyond fines, data breaches can lead to substantial financial losses due to breach recovery costs, customer compensation, and increased cybersecurity measures. A loss of consumer trust can also result in decreased sales and long-term revenue loss. Furthermore, privacy-conscious consumers are more likely to engage with companies that prioritize data protection. Businesses that fail to address privacy

concerns may find themselves at a competitive disadvantage, losing customers to more privacy-focused competitors.

In conclusion, algorithmic bias and privacy concerns have far-reaching implications for businesses. Addressing these issues is not only a legal and ethical obligation but also a strategic necessity to maintain consumer trust, operational efficiency, and competitive advantage. By fostering transparency, fairness, and robust data protection practices, businesses can navigate these challenges effectively and build a sustainable, trustworthy brand.

• Review practical1 and practical2 notebooks, look especially to the dataset's features, and summarize what are the independent variables that are sensible to bias (e.g., age, nationality, etc.), then list some methods that the literature suggests to minimize bias when manipulating these variables.

Independent Variables Susceptible to Bias

Housing Dataset

- Housing Median Age: Bias can occur if older housing areas are unfairly evaluated compared to newer ones.
- Median Income: Socio-economic bias could affect how different income groups are treated.
- Ocean Proximity: Geographic bias might affect areas differently based on their distance to the coast.

Customer Dataset

- Country, State, City, Zip Code: Geographic bias can impact how services or products are offered to customers in different regions.
- Gender: Gender bias can lead to unfair treatment or different service levels between male and female customers.
- Senior Citizen: Age-related bias can affect senior citizens differently compared to younger customers.
- Partner, Dependents: Family status bias can influence how services are marketed or provided.
- Churn Score, Churn Value, Churn Reason: These could introduce bias if not properly balanced across different demographic groups.

Method	**Description**
Data Collection	**Diverse and Representative Sample**: Ensure data covers different demographics and regions.
	Audit Data Collection Processes: Regularly check and correct biases in data collection.
Data Preprocessing	**Normalization and Standardization**: Scale features similarly to

	prevent dominance.
	Imputation: Carefully handle missing values to avoid introducing bias.
Bias Detection and Mitigation Techniques	**Fairness Metrics**: Use metrics like disparate impact, equal opportunity difference, and statistical parity difference to detect bias.
	Re-sampling Methods: Balance dataset by oversampling minority class or undersampling majority class.
	Adversarial Debiasing: Train models with adversaries to penalize biased outcomes.
	Fair Representation Learning: Learn representations invariant to sensitive features to reduce bias.
Algorithmic Adjustments	**Fair Algorithms**: Use fair algorithms such as Fair Logistic Regression, Fair SVM, or Fair k-means clustering.
	Regularization: Apply techniques to prevent overfitting to biased patterns.
Post-processing	**Bias Correction**: Adjust model outputs to ensure fairness across groups.
	Threshold Adjustment: Calibrate decision thresholds for fairness in outcomes.
Continuous Monitoring and Evaluation	**Monitor Models Regularly**: Continuously check models for bias and retrain as necessary.
	Bias Impact Assessments and Audits: Conduct regular assessments to ensure compliance with ethical standards and regulations.

Task 2

• Explore the practical3 jupyter notebook, fix 5 existing errors (e.g., semantic, logical, runtime, etc.), and submit notebook without errors.

Mistake	Solution
`sns.distplot(df[x], bins = 0)`	`sns.distplot(df[x], bins = 20)` `#bins can't be zero`
`KMeans init = 'kkk' parameter wrong`	`Init = 'k-means++' # according to comments`
`AttributeError: 'KMeans' object has`	`centroids1 =`

`no attribute 'centers_'`	`algorithm.cluster_centers_ #it's cluster_centers_ not centers_`
`C2. Implementing with k=6 # df['cluster'] = pd.DataFrame(y_kmeans)`	`y_kmeans already contains the labels from fitting the KMeans model:`
`When exporting to excel: ValueError: No engine for filetype: 'csv'`	`df.to_excel("datasets/customers segmented_final.xlsx", index = False)`

• Reflect on the notebook structure: Steps on data preprocessing, model implementation. What three top things you would do differently (better)? List what would you do differently and explain why.

- Modularizing the code more would help maintainability and management. Breaking the code into reusable functions or modules for data preprocessing, model fitting, and plotting would make the code more organized and easier to debug or extend.

- In the original notebook, there is a step to check for missing values, but no systematic approach to handle them if they are found. It's crucial to address missing values because they can significantly impact the performance of machine learning algorithms. By implementing a method to handle missing values, such as filling them with the mean, median, or using more sophisticated imputation techniques, we ensure the dataset's integrity and robustness. Additionally, validating data before proceeding with clustering helps in identifying any anomalies or errors early, ensuring that the analysis is based on clean and reliable data. This step is vital for improving the accuracy and reliability of the k-means clustering results.

- Visualization could be improved by using more advanced libraries like plotly,

By modularizing the code, handling missing values, and enhancing data visualization, the notebook will become more structured, robust, and insightful. These improvements will make the code easier to maintain and extend, and the visualizations will provide clearer insights into the data and the results of the clustering algorithm.

Task 3

In this task, you will explore the trade-off: processing time and number of features.

1. Select one notebook from either Lecture 1 or 2, and with that one dataset:

• Run the algorithms used in these notebooks with 1. double of number of features (use feature engineering for that), and 2.half number of features. To maximize the number of features, you can use feature engineering, and to reduce the number of features, you can do it randomly.

On the lecture 1 notebook:

I have used the PolynomialFeatures Library in sklearn to double the number of features, while for halving, I have just used a ransom sample from the original feature set.

```
from sklearn.preprocessing import PolynomialFeatures

# Create polynomial features
poly = PolynomialFeatures(degree=2, include_bias=False)
housing_poly = poly.fit_transform(housing_prepared)

# Convert to DataFrame for easier handling
housing_poly_df = pd.DataFrame(housing_poly, columns=poly.get_feature_names(housing_prepared.columns))
X_double = housing_poly_df
```

```
import random

# Randomly select half of the features
original_features = list(housing_prepared.columns)
reduced_features = random.sample(original_features, len(original_features) // 2)
X_half = housing_prepared[reduced_features]
```

Then I have just added some metrics to the training:

```
import time
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

def train_and_evaluate(X, y):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    model = LinearRegression()

    start_time = time.time()
    model.fit(X_train, y_train)
    end_time = time.time()

    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    processing_time = end_time - start_time

    return mse, r2, processing_time

# Double features
mse_double, r2_double, time_double = train_and_evaluate(X_double, housing_labels)

# Half features
mse_half, r2_half, time_half = train_and_evaluate(X_half, housing_labels)

print("Double Features Metrics:\nMSE: ", mse_double, "\nR2 Score: ", r2_double, "\nProcessing Time: ", time_double)
print("\nHalf Features Metrics:\nMSE: ", mse_half, "\nR2 Score: ", r2_half, "\nProcessing Time: ", time_half)
```

Which provided the following results:

```
Double Features Metrics:
MSE:  53517659843.89042
R2 Score:  -3.1174783213876873
Processing Time:  0.18309807777404785

Half Features Metrics:
MSE:  5195446960.705969
R2 Score:  0.6002788557454366
Processing Time:  0.002545595169067383
```

• Compare the validation metrics for each case and the processing time used in each case. Discuss the results.

	Performance	Processing Time
Double Features	The model with double the features has a very high MSE (53,517,659,843.89) and a negative R2 score (-3.12). A negative R2 score indicates that the model is performing worse than a horizontal line (mean of the target values). This suggests severe overfitting or that the additional features are introducing noise rather than useful information.	The processing time for the model with double the features is higher (0.1831 seconds), reflecting the increased complexity and computational cost due to the larger number of features.
Half Feature	The model with half the features has a much lower MSE (5,195,446,960.71) and a positive R2 score (0.60). This indicates that the model is fitting the data reasonably well and explaining 60% of the variance in the target variable, which is a good performance.	The model with half the features trains and predicts much faster (0.0025 seconds), demonstrating high computational efficiency.

In this specific scenario, the model with half the features is clearly the better choice due to its good predictive performance and superior computational efficiency. This highlights the importance of not just increasing the number of features but ensuring that the features added are relevant and contribute positively to the model's performance.

Generate an ML pipeline (a series of automated steps that are integrated into a single workflow for building and deploying ML models) for data preprocessing using ChatGPT (or any other LLM). This chunk of code should allow preprocessing data that contains continuous and categorical features. After, discuss the outputs, what's missing, and what could be done better.

The same dataet as assignmen 1 was used for convenience:

Pipeline

```
# Separate numerical and categorical columns
num_attribs = list(housing.drop("ocean_proximity", axis=1))
cat_attribs = ["ocean_proximity"]

# Numerical pipeline
num_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy="median")),
    ('std_scaler', StandardScaler()),
])

# Full pipeline for both numerical and categorical attributes
full_pipeline = ColumnTransformer([
    ("num", num_pipeline, num_attribs),
    ("cat", OneHotEncoder(), cat_attribs),
])

housing_prepared = full_pipeline.fit_transform(housing)

```

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(housing_prepared, housing_labels, test_size=0.2, random_state=42)

# Train the model
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

# Make predictions
y_pred = lin_reg.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("MSE: ", mse)
print("R2 Score: ", r2)

```

```
MSE:  4530316161.348168
R2 Score:  0.6514519013388278
```

What Could Be Improved:

- Additional feature engineering could be performed to create more meaningful features from the existing ones.
- Instead of using simple median imputation, more sophisticated methods like KNN imputation or model-based imputation could be tested.
- Other scaling methods, such as Min-Max scaling or robust scaling, could be explored.
- For categorical variables, techniques like target encoding or frequency encoding could be tested for potentially better performance.
- Implementing cross-validation to ensure the model generalizes well to unseen data.
- Hyperparameter tuning using grid search or random search to optimize the model's performance.