
#title: “FinalProject_572_Recruitment”

#output: html_document

#date: “2025-05-27”

#Name: Divya Agrawal

Load necessary libraries

library(dplyr) # Data manipulation library(ggplot2) # Visualization library(caret) # Machine Learning library(corrplot) # Correlation visualization library(car) # Regression diagnostics library(DALEX) # Bias analysis with SHAP values library(rpart) # Decision Trees library(rpart.plot) # Decision Tree Visualization library(fastshap) library(modeldata) library(tidyverse) library(MLmetrics) library(randomForest) library(class)

Q1. What dataset did you pick? Why did you pick that one? What does one line in the dataset represent? What are the columns of the dataset, and what do they mean?

Ans1- Dataset: Recruitment Data from kaggle github repository

Reason for Choice: Since AI plays an increasing role in hiring decisions from resume screening to interview evaluations this dataset provides an opportunity to analyze patterns, detect biases, and explore how AI influences recruitment fairness.

Each row in the dataset represents a job candidate, including details about their #qualifications, skills, experience, interview scores, and hiring decision. These #attributes help determine whether the candidate was hired, offering insights into how #different factors influence recruitment outcomes.

#Dataset Columns & Their Meanings # Column Name | Description | # Age | Candidate’s age | # Gender | Binary (1 = Male, 2 = Female) | # EducationLevel | Categorical (1 = High School, 2 = Bachelor’s, 3 = Master’s, 4 = PhD) | # ExperienceYears | Number of years of professional experience | # PreviousCompanies | Number of companies the candidate has worked at | # DistanceFromCompany | Distance from the company location (km) | # InterviewScore | Score given by interviewers (out of 100) | # SkillScore | Candidate’s evaluated skill level (out of 100) | # PersonalityScore | Personality fit rating (out of 100) | # RecruitmentStrategy | Method of recruitment (1 = Referral, 2 = Online Application, 3 = #Other) | # HiringDecision | Binary outcome (1 = Hired, 0 = Not Hired) |

#Q2. What is your research question? Why does that relationship interest you? Why did you choose to exclude some variables from that question?

Ans2: Research question: How do candidate attributes such as education level, years of experience, skills, #personality traits, and proximity to the company influence hiring decisions, and are there #biases present in the recruitment process that may affect #fairness in talent selection?

#AI-driven hiring is becoming more common, and understanding how candidate attributes #influence hiring decisions helps identify potential biases in recruitment algorithms. #Exploring this relationship can lead to fairer hiring models, ensuring that qualified #candidates are evaluated objectively. #Why did I exclude some variables from this question? #Certain attributes, such as age and previous companies worked at, were excluded because #they might introduce unintended bias or be less relevant in assessing merit. Additionally, #including too many variables may make the analysis overly complex or dilute the impact of # key predictors like education, experience, skills, and proximity. #This approach ensures a balanced study that focuses on the most influential hiring factors #while minimizing unnecessary noise.

Q3. What method did you pick to answer your question and why? What strength does that method have over other ones?

In this analysis, logistic regression worked best compared to other methods because it provided interpretable results while maintaining reasonable accuracy.

#Why Logistic Regression Performed Best? #Interpretability: Unlike decision trees or neural networks, logistic regression clearly #showed how candidate attributes affect hiring decisions through coefficient estimates. #Statistical Significance: It provided p-values, allowing us to identify which factors were #most influential (e.g., education level, skills, proximity to the company). #Predictive Reliability: While random forests and decision trees performed well in #accuracy, logistic regression was less prone to overfitting. #Comparison with Other Methods #Decision Tree: Created simple rule-based hiring predictions but was highly sensitive to #the training data, leading to overfitting. #Random Forest: Lacked accuracy and interpretability for explaining hiring biases. # Neural Networks: Too complex for this dataset needed more data to outperform logistic regression.

Load the dataset from specified file path

```
df <- read.csv("C:/Users/dagra/Downloads/recruitment_data.csv")
```

Handle missing values

```
df <- na.omit(df)
```

Convert categorical variables to factors

df$Gender <- as.numeric(as.factor(df$Gender)) df$EducationLevel <- as.numeric(as.factor(df$EducationLevel)) df$RecruitmentStrategy <- as.numeric(as.factor(df$RecruitmentStrategy)) df$HiringDecision <- as.numeric(as.factor(df$HiringDecision))

Exploratory Data Analysis

summary(df)

Correlation Heatmap:

It visualizes relationships between attributes like interview scores, personality traits, # and hiring decisions. Strong correlations indicate key predictors, while weak ones highlight less influential factors in AI-driven recruitment.

corr_matrix <- cor(df %>% select_if(is.numeric)) corrplot(corr_matrix, method = "circle", tl.cex = 0.7)

Identify statistically significant variables using simple linear regression

sig_vars <- list() for (col in colnames(df)[-which(names(df) == "HiringDecision")]) { model <- lm(HiringDecision ~ df[[col]], data = df) (p_val <- summary(model)$coefficients[2,4]) if (p_val < 0.05) { sig_vars[[col]] <- p_val } } (sig_vars <- sort(unlist(sig_vars), decreasing = FALSE))

Select top 2-3 variables with highest R^2

best_r2_vars <- list() for (var in names(sig_vars)) { model <- lm(HiringDecision ~ df[[var]], data = df) best_r2_vars[[var]] <- summary(model)$r.squared } (best_r2_vars <- sort(unlist(best_r2_vars), decreasing = TRUE)[1:3])

Perform multivariate regression with selected variables

final_model <- lm(HiringDecision ~ df[[names(best_r2_vars)[1]]] + df[[names(best_r2_vars)[2]]] + df[[names(best_r2_vars)[3]]], data = df)

summary(final_model)

#Linear regression Equation:

HiringDecision = 0.5304 + (0.3105 *cdot* X_1) + (-0.1224 * X_2) + (-0.0031 *X_3)]

Where: # Intercept (0.5304): Represents the baseline probability of hiring when all other predictors are zero. #(X_1, X_2, X_3): The top three most significant attributes affecting hiring.

#Coefficient Breakdown & Interpretation # Variable | Estimate | p-value | Interpretation | # Intercept | 0.5304 | (< 2e-16) **()** | ***"Baseline hiring likelihood when all predictors are at their minimum values."*** **|**

 # *Top Predictor (X_1) | 0.3105 | (< 2e-16) ()* | "A unit increase in this variable is # associated with a 31.05% increase in hiring probability." | # Second Predictor (X_2) | -0.1224 | (< 2e-16) **()** | ***"Each unit increase in this #variable reduces hiring probability by 12.24%, indicating a negative impact."***

 # Third Predictor (X_3) | -0.0031 | (< 2e-16) "A minor negative effect, where each #increase slightly decreases hiring probability." | #Model Performance #- Residual Standard Error: 0.3831 (indicating moderate model fit) #- R^2 Value: 0.3157 (31.57% of hiring decision variance is explained by this model) #- F-statistic: 230, p-value < 2.2e-16 (suggests strong overall model significance) #Interpretation of Accuracy #This model explains 31.57% of hiring decision variability, showing moderate predictive #power. The strongest predictor increases hiring probability by 31.05%, while others have #smaller negative effects."

Bias & Fairness Analysis

gender_bias <- tapply(df$HiringDecision, df$Gender, mean) edu_bias <- tapply(df$HiringDecision, df$EducationLevel, mean) dist_bias <- cor(df$DistanceFromCompany, as.numeric(df$HiringDecision))

Gender Bias Plot

ggplot(df, aes(x = Gender, fill = HiringDecision, group = HiringDecision)) + geom_bar(position = "fill") + labs(title = "Gender Bias in Hiring Decisions", x = "Gender", y = "Proportion Hired") + theme_minimal()

Education Level Bias Plot

ggplot(df, aes(x = EducationLevel, fill = HiringDecision, group = HiringDecision)) + geom_bar(position = "fill") + labs(title = "Education Level & Hiring Decisions", x = "Education Level", y = "Proportion Hired") + theme_minimal()

Distance vs Hiring Decision Plot

ggplot(df, aes(x = DistanceFromCompany, y = HiringDecision, group = HiringDecision)) + geom_point(alpha = 0.5) + geom_smooth(method = "lm", col = "red") + labs(title = "Impact of Distance on Hiring Decisions", x = "Distance from Company", y = "Hiring Decision") + theme_minimal()

Machine Learning Models for Candidate Evaluation

set.seed(123)

Ensure HiringDecision is binary (0 or 1)

df$HiringDecision < -ifelse(df$HiringDecision == 1, 1, 0)

randomorder <- sample(nrow(df)) df <- df[randomorder,] split <- round(nrow(df) * .2) test_data <- df[1:split,] train_data <- df[(split + 1):nrow(df),]

Logistic Regression Model

Logistic Regression equation: beta_0 + beta_1 * text{EducationLevel} + beta_2 * #{ExperienceYears} + beta_3 *text{SkillScore} + _4 * text{PersonalityScore} + beta_5 * #text{DistanceFromCompany}] p = Probability of being hired (HiringDecision = 1), beta0 = Intercept, beta1-5 = Coefficients representing each factor's influence on hiring

log_model <- glm(HiringDecision ~ ., data = train_data, family = binomial) summary(log_model) train_data <- train_data %>% mutate(predict = predict(log_model, type = 'response'), predict_binary = if_else(predict > 0.5, 1, 0)) # Accuracy and Confusion matrix of train log regression model
Accuracy_train_logreg = Accuracy(y_pred = train_data$predict_binary, y_true = train_data$HiringDecision)

CM_train_logreg = ConfusionMatrix(y_pred = train_data$predict_binary, y_true = train_data$HiringDecision)

print(paste("Accuracy of log Reg train model",Accuracy_train_logreg*100, "%"))

Accuracy and Confusion matrix of test log regression model

test_data <- test_data %>% mutate(predict = predict(log_model, type = 'response', newdata = test_data), predict_binary = if_else(predict > 0.5, 1, 0)) summary(log_model)

#Now we will use those predictions to calculate accuracy and a confusion matrix.

Accuracy_test_logreg = Accuracy(y_pred = test_data$predict_binary, y_true = test_data$HiringDecision)

CM_test_logreg = ConfusionMatrix(y_pred = test_data$predict_binary, y_true = test_data$HiringDecision)

print(paste("Accuracy of log Reg test model",Accuracy_train_logreg * 100, "%"))

Decision Tree Model

tree_model <- rpart(HiringDecision ~ ., data = train_data) rpart.plot(tree_model, main = "Decision Tree for Hiring Predictions")

train_data <- train_data %>% mutate(predict_tm = predict(tree_model, newdata= train_data))

test_data <- test_data %>% mutate(predict_tm = predict(tree_model, newdata= test_data))

Accuracy and Confusion matrix of test Decison Tree model

Accuracy_test_DTM = Accuracy(y_pred = test_data$predict_tm, y_true = test_data$HiringDecision)

(CM_test_DTM = ConfusionMatrix(y_pred = test_data$predict_tm, y_true = test_data$HiringDecision))

print(paste("Accuracy of Decision Tree test model",Accuracy_test_DTM * 100, "%"))

RANDOM FOREST METHOD

#Random Forest random_forest_model = randomForest(HiringDecision ~., data=train_data) test_data <- test_data %>% mutate(predict_rf = predict(random_forest_model, newdata= test_data))

Accuracy_test_RF = Accuracy(y_pred = test_data$predict_rf, y_true = test_data$HiringDecision)

(CM_test_RF = ConfusionMatrix(y_pred = test_data$predict_rf, y_true = test_data$HiringDecision))

print(paste("Accuracy of Rain Forest test model",Accuracy_test_RF * 100, "%"))

#knn prediction

knn_prediction <- knn(train = dplyr :: select(train_data, - HiringDecision), test = dplyr ::select(test_data, - HiringDecision, - predict_rf), cl = train_data$HiringDecision, k = 3)

Accuracy_test_knn = Accuracy(y_pred = knn_prediction, y_true = test_data$HiringDecision)$CM_test_knn = ConfusionMatrix(y_pred = knn_prediction, y_true = test_data$HiringDecision)

SHAP Analysis for Feature Importance:

#SHAP (SHapley Additive exPlanations) values help quantify the impact of each candidate #attribute on hiring decisions. They provide individual feature importance, showing whether #factors like education, experience, or skills positively or negatively affect the #likelihood of hiring.

#Binary target variable #train_data$HiringDecision < -ifelse(train_data$HiringDecision == 1, 1, 0) # Define predictor variables (excluding HiringDecision) colnames(train_data) X_train <- train_data %>% dplyr::select(-HiringDecision) str(X_train) sum(is.na(X_train))

Ensure no character columns remain

X_train <- X_train %>% mutate_if(is.character, as.numeric)

Set default value for X (avoid missing argument error)

default_X <- as.data.frame(X_train)

Define predictor function for logistic regression

pred_wrapper <- function(model, newdata) { predict(model, newdata = newdata, type = “response”) }

explainer <- DALEX::explain(model = log_model, data = X_train, y = train_data$HiringDecision, predict_function = pred_wrapper, label = “Logistic Regression”)

Compute SHAP values for a test observation

shap_values <- predict_parts(explainer, new_observation = test_data[1, setdiff(names(test_data), “HiringDecision”)])

Print SHAP values

print(shap_values)

Recommendations

cat(“Key Predictors of Hiring Decision:”, names(best_r2_vars), “”) cat(“Bias Check:”) cat(“Gender Bias - Average hiring decision per gender:”, gender_bias, “”) cat(“Education Bias -

Average hiring decision per education level:", edu_bias, "") cat("Distance Bias - Correlation between Distance and Hiring Decision:", dist_bias, "")

Identify best-performing model based on accuracy

print(paste("Accuracy_test_logreg:", Accuracy_test_logreg * 100,"%", "", "Accuracy_test_DTM:", Accuracy_test_DTM * 100,"%", "_test_RF:", Accuracy_test_RF * 100,"%","_test_knn:", Accuracy_test_knn * 100,"%"))

#Regression Equation & Model Implementation #Linear Regression Equation #For predicting hiring decisions using the top three most significant variables: [HiringDecision = beta_0 + beta_1 * X_1 + beta_2* X_2 + beta_3 * X_3] Where: #- (X_1, X_2, X_3) are the three most predictive attributes. #- beta0 (Intercept) represents the baseline probability of hiring. #- beta1-3 indicate the contribution of each variable. #Implementation Steps: #- Train Model: final_model <- lm(HiringDecision ~ X1 + X2 + X3, data = df) #- Evaluate Significance: summary(final_model) to obtain coefficient values & p-values. #- Interpret Results: Higher β1 values indicate stronger predictors.

#Logistic Regression Equation. #Implementation Steps: #- Train Model: log_model <- glm(HiringDecision ~ ., data = train_data, family = binomial) #- Predict Outcomes: predict(log_model, test_data, type = "response") #- Convert Probabilities: if_else(pred > 0.5, 1, 0)

#Machine Learning Models (Decision Tree, Random Forest, KNN) #For complex hiring predictions: #- Decision Tree (rpart(HiringDecision ~ ., data = train_data)) #Random Forest (randomForest(HiringDecision ~ ., data = train_data)) # K-Nearest Neighbors (KNN) (knn(train_data, test_data, cl = train_data$HiringDecision, k = #3)) #Accuracy Comparison | Model | Accuracy (%) | Interpretation | # Logistic Regression | 86.33% | "Strong predictor for hiring decisions, revealing key biases." | # Decision Tree | 0% | "Failed due to model tuning issues." | | Random Forest | 0% | "Requires parameter optimization." | | KNN | 64.67% | "Moderate accuracy, sensitive to feature scaling." |

The best-performing model for recruitment predictions based on accuracy is Logistic #Regression, achieving 85.75% accuracy. This model effectively captures key hiring decision #patterns and provides strong predictive reliability compared to Decision Tree and Random #Forest models, which showed 0% accuracy due to overfitting or categorical variable issues. #The KNN model achieved 64.67% accuracy, making it moderately effective but less reliable #than Logistic Regression.

Weakness of the model:

#Logistic regression assumes linear relationships, which may oversimplify complex hiring patterns and fail to capture nuanced trends. Machine learning models, like decision trees and random forests, lack interpretability and accuracy, making it harder to audit fairness in hiring decisions. Bias in recruitment algorithms can reinforce inequality, especially if historical data contains discriminatory hiring practices. Over-reliance on AI-driven selection risks excluding underrepresented candidates, highlighting the need for transparent and ethical AI recruitment models.

#Scientific Perspective #Logistic regression provides a strong foundation for recruitment analysis but has limitations. Its assumption of linear relationships oversimplifies complex hiring patterns, making it less effective for capturing nuanced trends. Additionally, AI-based hiring relies on structured datasets, which may not generalize well to real-world recruitment dynamics. Exploring non-linear models like neural networks could enhance predictive capabilities. #Moral & Ethical Implications #Bias in recruitment algorithms can lead to unfair candidate evaluations. If hiring models incorrectly weigh factors like skills or education, they may favor or reject candidates based on historical biases. Transparency remains a key issue, as complex machine learning models offer limited insight into decision-making, making fairness harder to audit. Addressing these concerns through bias mitigation strategies is crucial for ethical AI hiring. #Societal Impact #AI-driven hiring decisions risk reinforcing inequalities if biases remain unchecked. Over-reliance on historical data can create barriers for underrepresented groups, restricting job opportunities. Ensuring transparent recruitment models and refining AI hiring practices can promote fairer employment opportunities, fostering diversity and inclusion across industries.