PROGRAMMING FOR DATA ANALYTICS AND AI

Student ID number	Title of degree studying	Year
2301022	MSC ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING	2024

Short unit name:	M33879			Due date: 16 Sep 2024		
Full unit name:	PROGRAMMING FOR DATA ANALYTICS AND AI					
Unit lecturer name:	Mr. Johnson Ang			johnson.ang@kaplan.com		
Additional items e.g., CD/disk/USB:	Yes		No	✓	**Details:**	

All additional items should be clearly labelled with ID number and unit name and securely attached to your work.

Candidates are reminded that the following are defined as Assessment Offences and will be dealt with in accordance with theUniversity's Code of Student Discipline:

a) Any attempt to complete an assessment by means considered to be unfair;
b) Plagiarism, which the University defines as the incorporation by a student in work for assessment of material which is not their own, in the sense that all or a substantial part of the work has been copied without any adequate attempt at attribution or has been incorporated as if it were the student's own work when in fact it is wholly or substantially the work of another person or persons.

Please note: Group **coursework** will be filed under the first Student ID number at the top of the list. Ensure you know all **group member's ID numbers.**

NB: **Coursework not collected** will be disposed of six months **after** the hand-in date.

FOR OFFICIAL USE ONLY

Date received/Office stamp	Provisional mark % / Comments
Administration Office	Academic Staff Member

Predictive Analysis of Workforce Adaptation to COVID-19 Pandemic-Induced Remote Work

An In-Depth Analysis of the Emerging Trends and Behavioural Responses to the Transition of Academic and Professional Commitments into Virtual Environments During and Post-Pandemic using Machine Learning Models

I. ABSTRACT

The onset of the COVID-19 pandemic in March 2020 prompted governments worldwide to impose lockdowns, leading to a profound shift in how people live and work. One of the most significant changes was the widespread transition of both professional and academic activities to virtual platforms, replacing the traditional in-person interactions. This sudden and comprehensive shift to remote work and study brought with it a variety of challenges and opportunities, impacting individuals in diverse ways based on their unique circumstances.

To investigate these impacts, the authors designed a comprehensive questionnaire informed by the principles of self-efficacy theory. This survey was distributed to a broad demographic, encompassing students, working professionals, entrepreneurs, and homemakers aged between 12 and 60 years. The goal was to capture a wide range of experiences and perceptions related to the shift to remote environments. The data collected from these used to explore how various aspects of remote work—such as isolation, flexibility, and technology use—affect the mental well-being and productivity of different segments of the population, considering their specific professional or academic contexts.

Building on these insights, the study employed advanced machine learning algorithms to develop predictive models that classify individuals' preferences for remote or in-person work. These models aim to identify key factors that influence whether a person is more likely to thrive in a remote setting or prefers the structure of in-person engagements, ***gender wise (male/female),*** providing valuable insights for employers, educators, and policymakers as they navigate the ongoing evolution of work and learning environments in a post-pandemic world.

II. KEYWORDS

COVID-19, Supervised Learning, Classification Models, Regression Models, Clustering Model, Machine Learning, Exploratory Data Analysis (EDA), Correlation, Plotly, Joblib, MeanShift, Hierarchical Clustering, Heatmap, Silhouette, RoC Curve, Confusion Matrix, Jaccard, GridSearchCV

III. INTRODUCTION

COVID-19 rapidly spread across the globe, prompting the World Health Organization to officially declare it a pandemic in March 2020 (World Health Organization, 2020). As of October 2021, the virus had infected over 238 million people worldwide, with a death toll exceeding 4.8 million (Worldometers.info, 2021). In an effort to curb the virus's spread, governments imposed widespread lockdowns, which led to a massive shift towards remote work and online education.

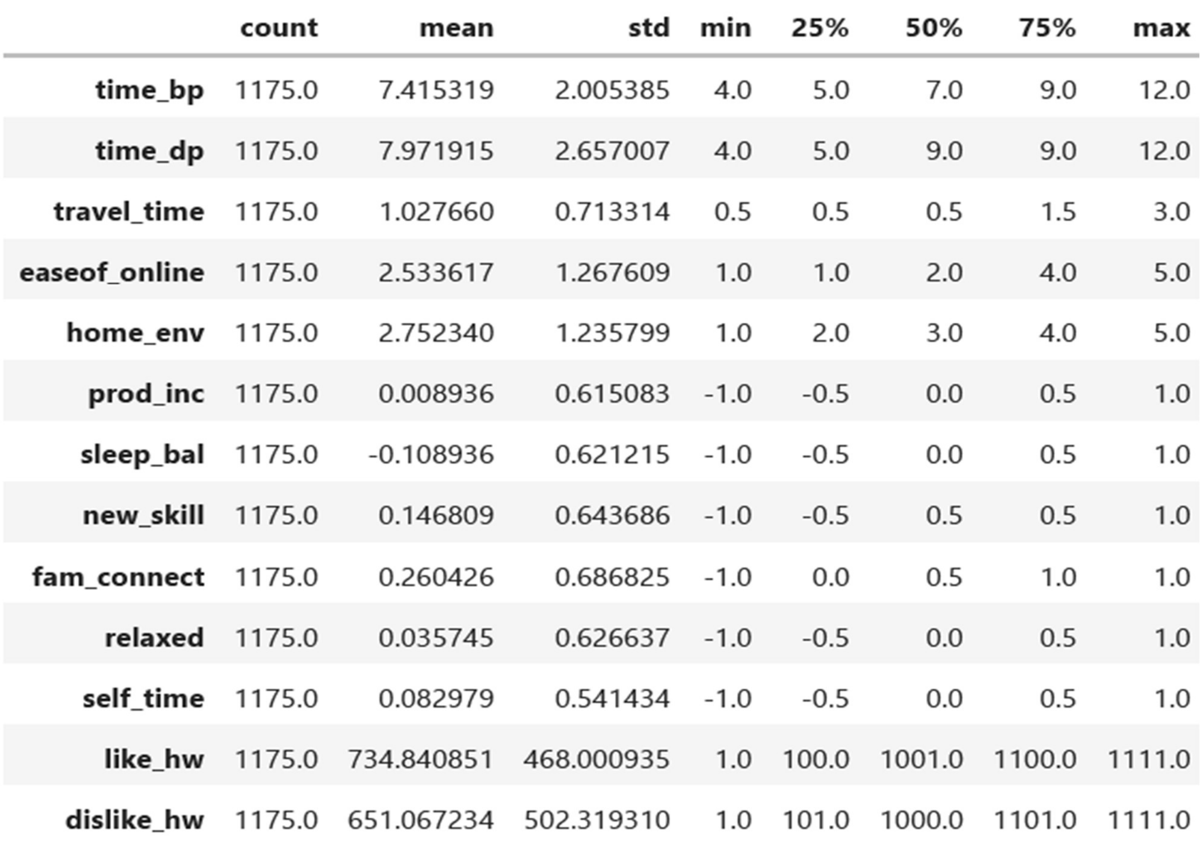
This study aims to explore the impact of complete remote work by analysing survey responses mostly aged 12 to 60 across India, all of whom were working from home full-time at the time of data collection. Participation in the study was voluntary, anonymous, and without any incentives. The research adhered to the ethical guidelines outlined in the Declaration of Helsinki (World Medical Association Declaration of Helsinki, 2013). The survey was designed based on the constructs of self-efficacy theory, as detailed in Section II, to assess factors influencing an individual's work performance, mental state, and overall experience in a virtual setting. The collected data was analysed using many plots and graphs including tables and then feature engineering being carried out to improve the data for analysis and predictive modelling further. Additionally, ***major supervised learning models were employed in predictive modelling to determine participants gender-male and female for preferred location either remote or in-person***.

IV. DATA LOADING AND REVIEW

All required libraries have been imported and the data is loaded into Dataframe object using panda's library and basic data review performed as first step.

V. EXPLORATORY DATA ANALYSIS

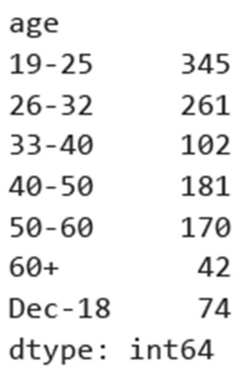
1. **Generate descriptive statistics**
   Descriptive statistics include those that summarize the central tendency, dispersion and shape of a dataset's distribution, excluding NaN values.



	count	mean	std	min	25%	50%	75%	max
time_bp	1175.0	7.415319	2.005385	4.0	5.0	7.0	9.0	12.0
time_dp	1175.0	7.971915	2.657007	4.0	5.0	9.0	9.0	12.0
travel_time	1175.0	1.027660	0.713314	0.5	0.5	0.5	1.5	3.0
easeof_online	1175.0	2.533617	1.267609	1.0	1.0	2.0	4.0	5.0
home_env	1175.0	2.752340	1.235799	1.0	2.0	3.0	4.0	5.0
prod_inc	1175.0	0.008936	0.615083	-1.0	-0.5	0.0	0.5	1.0
sleep_bal	1175.0	-0.108936	0.621215	-1.0	-0.5	0.0	0.5	1.0
new_skill	1175.0	0.146809	0.643686	-1.0	-0.5	0.5	0.5	1.0
fam_connect	1175.0	0.260426	0.686825	-1.0	0.0	0.5	1.0	1.0
relaxed	1175.0	0.035745	0.626637	-1.0	-0.5	0.0	0.5	1.0
self_time	1175.0	0.082979	0.541434	-1.0	-0.5	0.0	0.5	1.0
like_hw	1175.0	734.840851	468.000935	1.0	100.0	1001.0	1100.0	1111.0
dislike_hw	1175.0	651.067234	502.319310	1.0	101.0	1000.0	1101.0	1111.0

Table. *Descriptive statistics on given dataset*

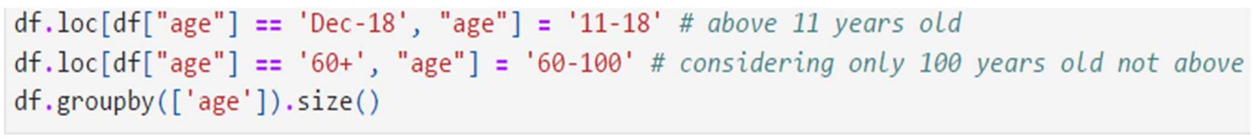
2. **Age Distribution and Assumptions Made on Age feature**



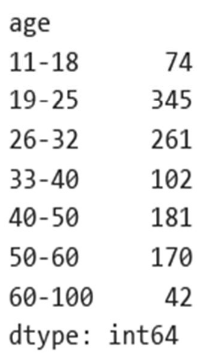
```
age
19-25      345
26-32      261
33-40      102
40-50      181
50-60      170
60+         42
Dec-18      74
dtype: int64
```

Table: *Age Distribution Count as per original dataset*

The values like 60+ and Dec-18 have been replaced with range values to have consistency as shown.



```
df.loc[df["age"] == 'Dec-18', "age"] = '11-18' # above 11 years old
df.loc[df["age"] == '60+', "age"] = '60-100' # considering only 100 years old not above
df.groupby(['age']).size()
```

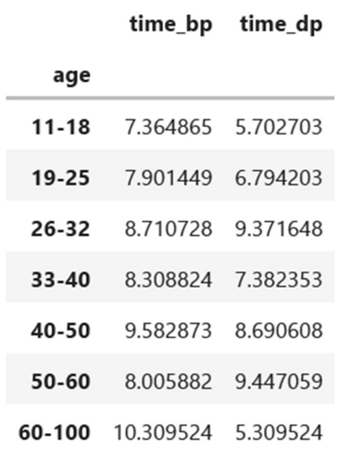


```
age
11-18       74
19-25      345
26-32      261
33-40      102
40-50      181
50-60      170
60-100      42
dtype: int64
```

3. **Daily Time Commitment**

Table below presents age-wise trends in the average time spent on work per day. The 11-18 age group shows a decrease of 1.66 hours in daily time commitment working from home, likely due to the elimination of travel time and reduced time spent preparing for school. A similar pattern is observed in the 19-25, 33-40, 40-50, 60-100 age groups, as shown in Table. Another significant factor contributing to the decline in work hours could be due widespread loss of employment during the pandemic.

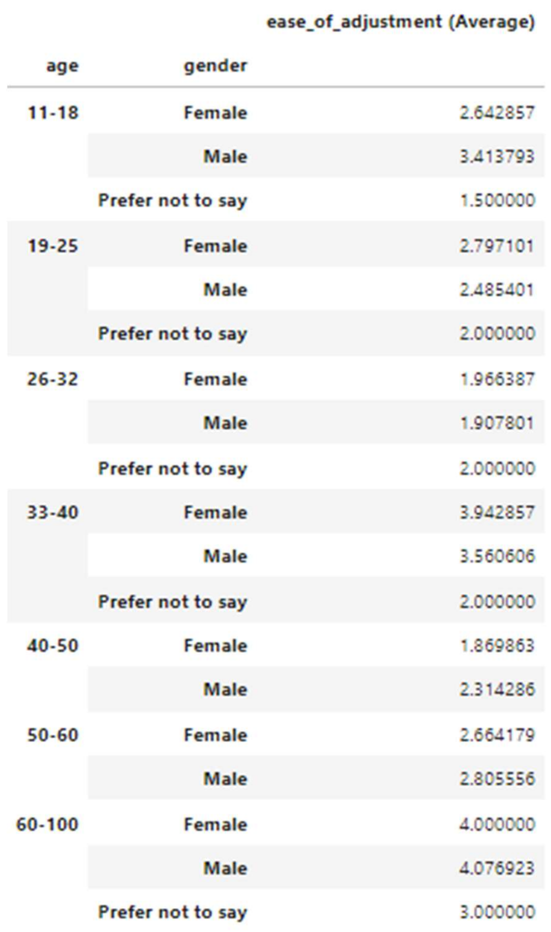
In contrast, respondents aged between 26-32 and 50-60 reported spending more time working from home than they previously spent on both commuting and working on a regular day. This trend extends beyond students, as most other demographics also reported an increase in working hours at home. Notably, homemakers experienced the most substantial rise in their working hours during the pandemic.



	time_bp	time_dp
age		
11-18	7.364865	5.702703
19-25	7.901449	6.794203
26-32	8.710728	9.371648
33-40	8.308824	7.382353
40-50	9.582873	8.690608
50-60	8.005882	9.447059
60-100	10.309524	5.309524

Table. *Daily time commitment across ages before and during the pandemic*

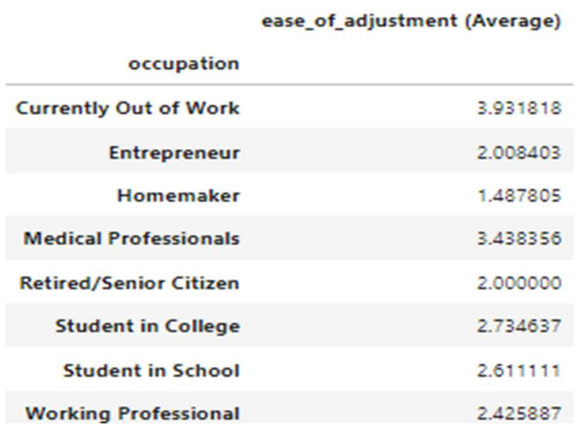
4. **Adjusting to New Setting by Gender**
   Respondents rated their ease of adjusting to an online environment on a scale from 1 to 5, with 1 indicating the least difficulty and 5 indicating the most. As shown in Table, males generally found the transition to an online environment more challenging than females, with the exception of those aged 19-25, 26-32 and 33-40. Among both genders, the 26-32 age group found it the easiest to adapt to online platforms. This age group typically corresponds to the early stages of a professional career, suggesting that individuals in the early career phase are more adept at navigating online work environments. Conversely, those aged 33-40 faced the greatest difficulty in adjusting, regardless of gender.



		ease_of_adjustment (Average)
age	**gender**	
11-18	**Female**	2.642857
	Male	3.413793
	Prefer not to say	1.500000
19-25	**Female**	2.797101
	Male	2.485401
	Prefer not to say	2.000000
26-32	**Female**	1.966387
	Male	1.907801
	Prefer not to say	2.000000
33-40	**Female**	3.942857
	Male	3.560606
	Prefer not to say	2.000000
40-50	**Female**	1.869863
	Male	2.314286
50-60	**Female**	2.664179
	Male	2.805556
60-100	**Female**	4.000000
	Male	4.076923
	Prefer not to say	3.000000

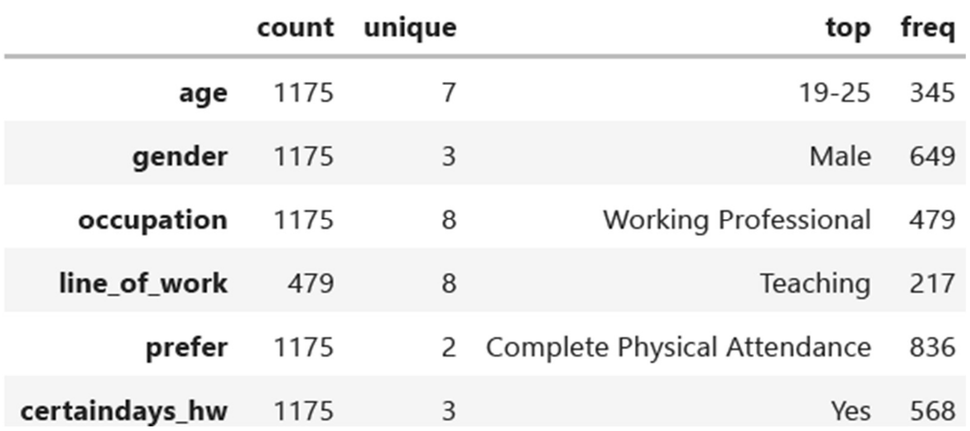
Table. *Trends across gender and age pertaining to the ease in adjusting to an online setting*

5. **Adjusting to New Setting by Occupation**



occupation	ease_of_adjustment (Average)
Currently Out of Work	3.931818
Entrepreneur	2.008403
Homemaker	1.487805
Medical Professionals	3.438356
Retired/Senior Citizen	2.000000
Student in College	2.734637
Student in School	2.611111
Working Professional	2.425887

6. **Tabular Distribution of categorical data**



	count	unique	top	freq
age	1175	7	19-25	345
gender	1175	3	Male	649
occupation	1175	8	Working Professional	479
line_of_work	479	8	Teaching	217
prefer	1175	2	Complete Physical Attendance	836
certaindays_hw	1175	3	Yes	568

7. **Preference for Attendance Mode – Physical/Home**
The size of each segment reflects the relative preference of respondents for each mode of attendance, offering insights into the overall inclination of the group towards either physical or home-based attendance. Preference is more for physical attendance

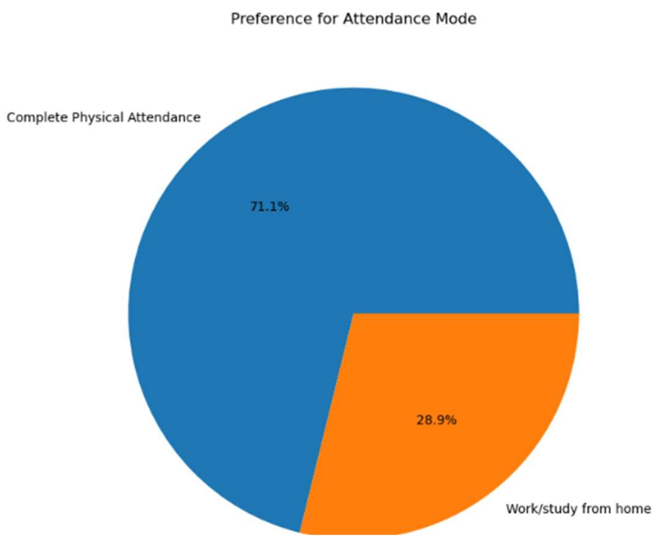


Figure. *Distribution of attendance mode preference*

8. **Ease of Online Working**
Respondents rated their ease of online working while working remote from 1 to 5 in increasing order. The chart shows respondents feel ease of online working more compared to physical presence.

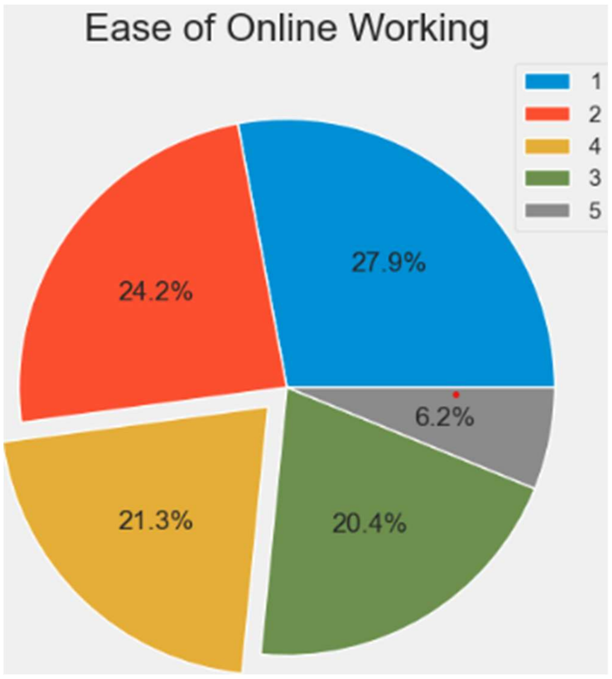


Figure. *Distribution of ease of online working level*

9. **Home Environment levels**
Respondents rated their home environment level while working online/from home
from 1 to 5 in increasing order. The chart shows respondents feel having good home environment on an average.

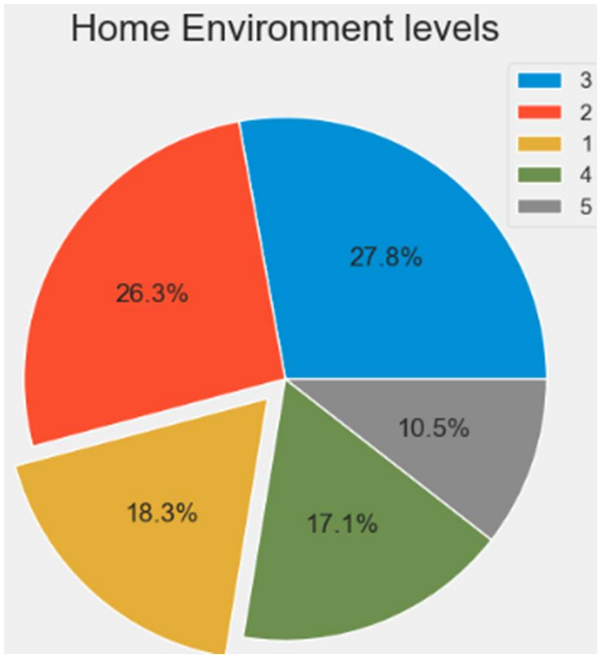
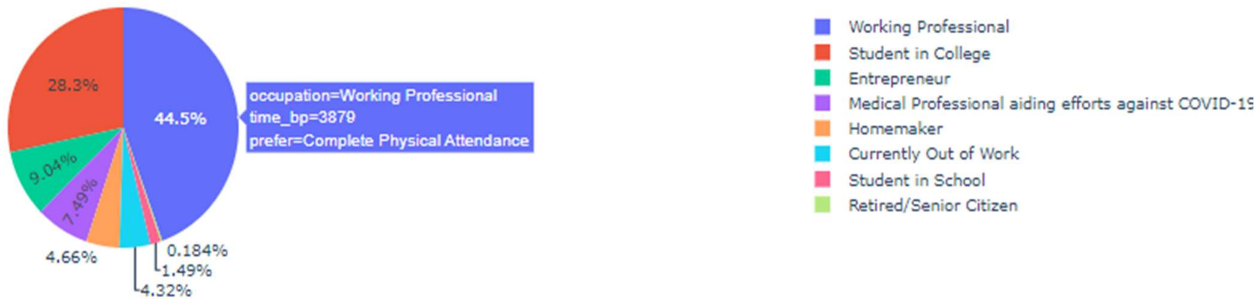


Figure. *Distribution of Preference for Attendance Mode – Physical/Homer*

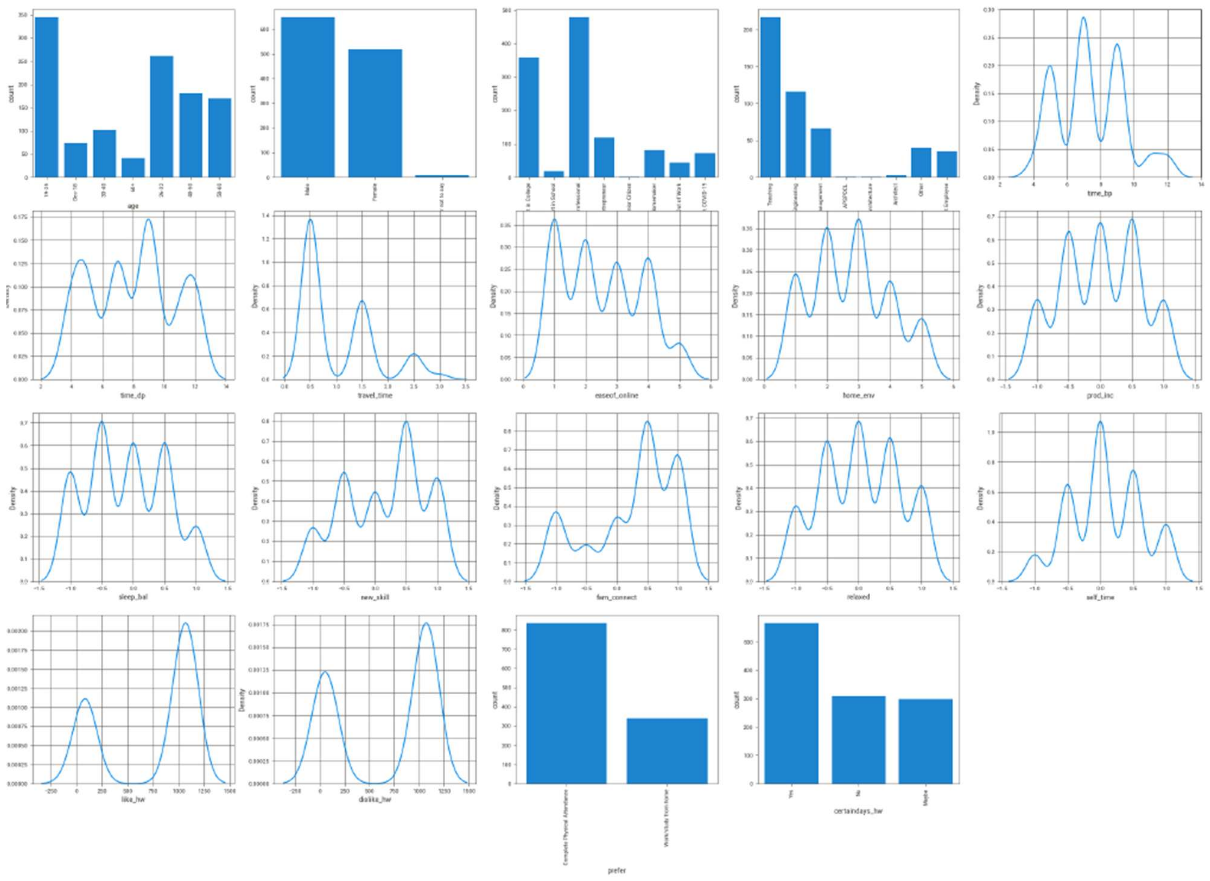
10. Distribution of Occupation using Plotly

Plotly Express is a better option for EDA than Seaborn. Each Library has its own advantages, But Plotly is a clear winner when it's come to interactive graphs which allow you to dig in to deeper insights during your EDA.



11. Probability density curve using kernel density estimate (KDE) plot

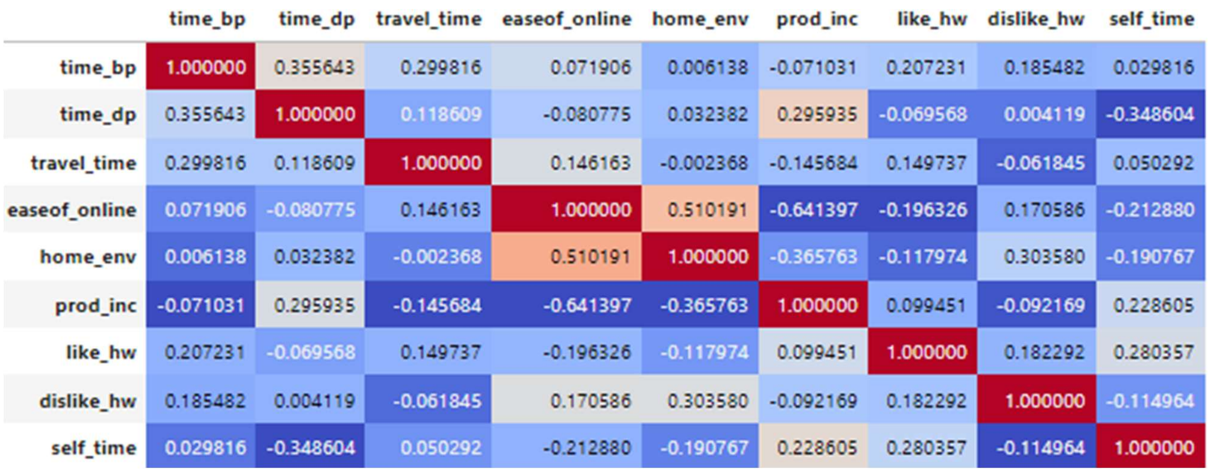
A kernel density estimate (KDE) plot is a method for visualizing the distribution of observations in a dataset, analogous to a histogram. KDE represents the data using a continuous probability density curve in one or more dimensions



12. Bivariate correlation matrix

Correlation is a bivariate analysis that measures the strength of association between two variables and the direction of the relationship. Used most widely used correlation - Pearson correlation to measure the degree of the relationship between linearly related variables. Other correlation measures that can be used are Kendall rank correlation, Spearman correlation, and the Point-Biserial correlation.

This correlation matrix shows that there are no pairs of features are correlated among the measured features as shown.



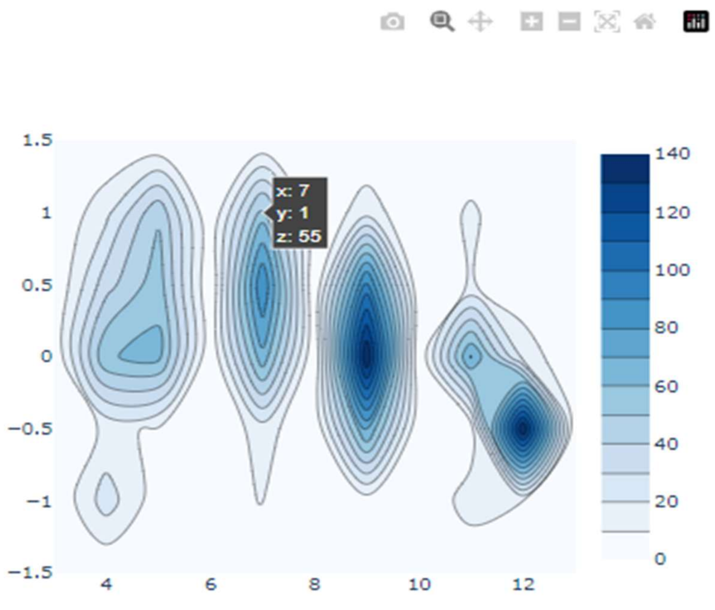
	time_bp	time_dp	travel_time	easeof_online	home_env	prod_inc	like_hw	dislike_hw	self_time
time_bp	1.000000	0.355643	0.299816	0.071906	0.006138	-0.071031	0.207231	0.185482	0.029816
time_dp	0.355643	1.000000	0.118609	-0.080775	0.032382	0.295935	-0.069568	0.004119	-0.348604
travel_time	0.299816	0.118609	1.000000	0.146163	-0.002368	-0.145684	0.149737	-0.061845	0.050292
easeof_online	0.071906	-0.080775	0.146163	1.000000	0.510191	-0.641397	-0.196326	0.170586	-0.212880
home_env	0.006138	0.032382	-0.002368	0.510191	1.000000	-0.365763	-0.117974	0.303580	-0.190767
prod_inc	-0.071031	0.295935	-0.145684	-0.641397	-0.365763	1.000000	0.099451	-0.092169	0.228605
like_hw	0.207231	-0.069568	0.149737	-0.196326	-0.117974	0.099451	1.000000	0.182292	0.280357
dislike_hw	0.185482	0.004119	-0.061845	0.170586	0.303580	-0.092169	0.182292	1.000000	-0.114964
self_time	0.029816	-0.348604	0.050292	-0.212880	-0.190767	0.228605	0.280357	-0.114964	1.000000

Figure. *Bivariate correlation matrix among measured features*

13. 2D Histogram Contours or Density Contours with Plotly express

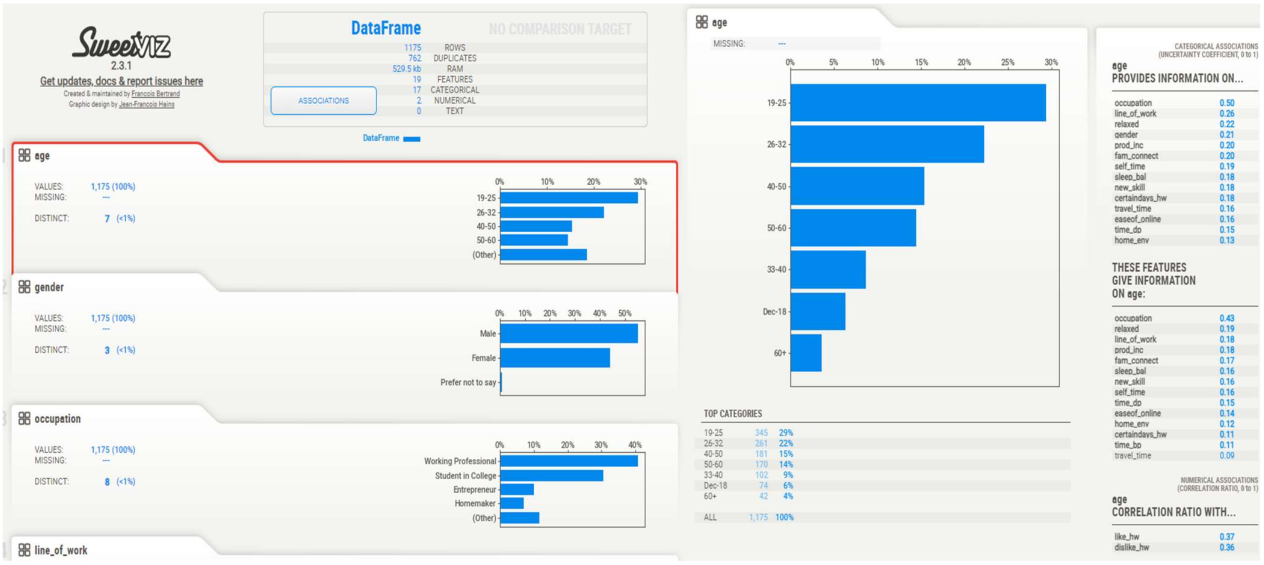
Density contours are lines that represent areas of equal data point density, similar to how a topographic map uses contour lines to show areas of equal elevation. In the context of a 2D histogram, density contours can help visualize the concentration of data points across the two variables, highlighting regions where data points are densely packed.
Here one can visualize how two variables, such as "time_dp" and "self_time" are related across given dataset. The one high density region in the plot is where time_dp is around 9 hours and respondent getting no or quit little time for self time.

```
fig = go.Figure(go.Histogram2dContour( x = df['time_dp'], y = df['self_time'], colorscale = 'Blues'
))
fig.update_layout(width=500, height=500)
fig.show()
```



14. SweetViz | Automated Exploratory Data Analysis (EDA)

Sweetviz is used which is an open-source Python library that generates beautiful, high-density visualizations to kickstart EDA (Exploratory Data Analysis). Its goal is to help quick analysis of target characteristics, training vs testing data, and other such data characterization tasks.



DataFrame	
1175	ROWS
762	DUPLICATES
529.5 kb	RAM
19	FEATURES
17	CATEGORICAL
2	NUMERICAL
0	TEXT

age	
VALUES:	1,175 (100%)
MISSING:	---
DISTINCT:	7 (<1%)

gender	
VALUES:	1,175 (100%)
MISSING:	---
DISTINCT:	3 (<1%)

occupation	
VALUES:	1,175 (100%)
MISSING:	---
DISTINCT:	8 (<1%)

age

MISSING: ---

TOP CATEGORIES

19-25	345	29%
26-32	261	22%
40-50	181	15%
50-60	170	14%
33-40	102	9%
Dec-18	74	6%
60+	42	4%
ALL	1,175	100%

CATEGORICAL ASSOCIATIONS (UNCERTAINTY COEFFICIENT, 0 to 1)

age PROVIDES INFORMATION ON...

occupation	0.50
line_of_work	0.26
relaxed	0.22
gender	0.21
prod_inc	0.20
fam_connect	0.20
self_time	0.19
sleep_bal	0.18
new_skill	0.18
certaindays_hw	0.18
travel_time	0.16
easeof_online	0.16
time_dp	0.15
home_env	0.13

THESE FEATURES GIVE INFORMATION ON age:

occupation	0.43
relaxed	0.19
line_of_work	0.18
prod_inc	0.18
fam_connect	0.17
sleep_bal	0.16
new_skill	0.16
self_time	0.16
time_dp	0.15
easeof_online	0.14
home_env	0.12
certaindays_hw	0.11
time_bp	0.11
travel_time	0.09

NUMERICAL ASSOCIATIONS (CORRELATION RATIO, 0 to 1)

age CORRELATION RATIO WITH...

like_hw	0.37
dislike_hw	0.36

VI. Data Preparation and Feature Engineering

1. **Handling Missing Data**
   It has been found that around 59.23 % of data was missing from the feature: line_of_work so has been containing null so has been dropped. Did not perform the imputation of missing values with some other value to avoid any kind of deviation in the results and also this feature was not adding any value.

2. **Handling Duplicate Data**
   The dataset is free of duplicate values because all questions were mandatory, and respondents were allowed to complete the survey only once.

3. **Handling Feature based on combination of other features**
   One aspect of feature engineering is the creation of new features out of existing features. A simple example would be to create a new feature which is the sum of the number of bathrooms in the house: df['time_bp'] = df['time_bp'] + df['travel_time']

4. **Outlier Detection**
   BoxPlot is plotted to see if there any outliers in the features and found probable outlier in time_bp feature, probably due do feature engineering that is performed by summed up the 'travel_time'.

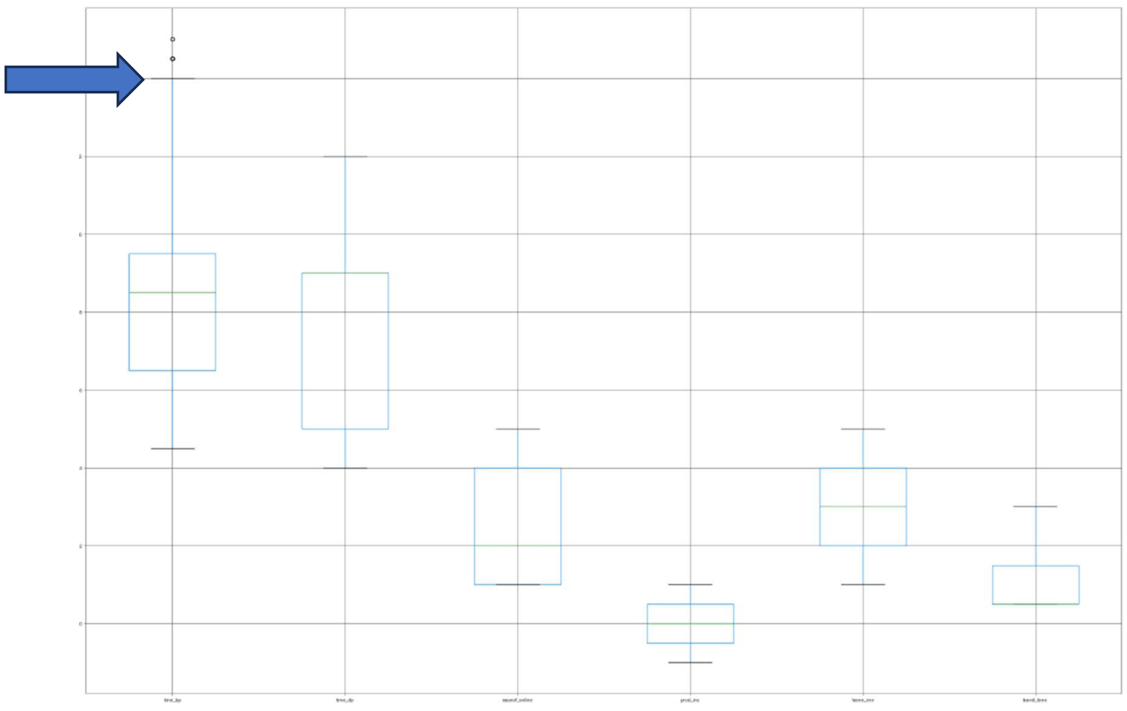
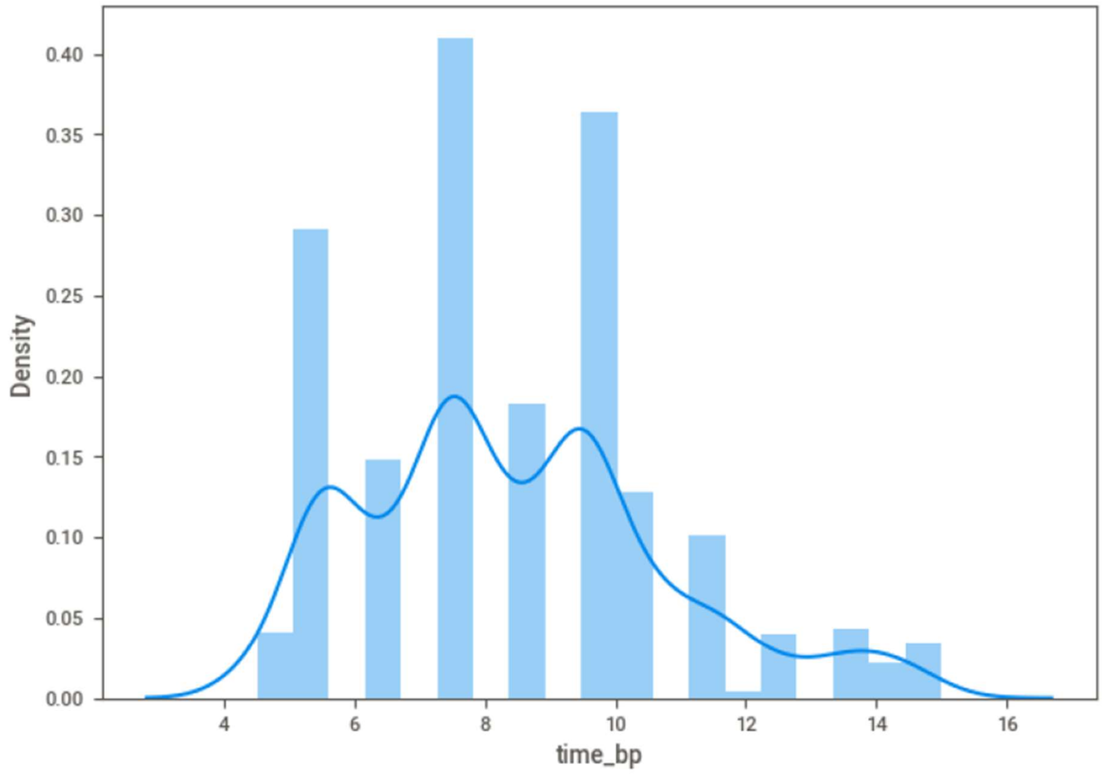


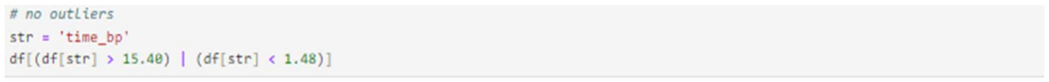
Figure. *Boxplot showing probable outlier in time_bp feature*

To investigate further tried plotting distplot and set the lower limit (lower_limit) to be three standard deviations below the mean, and the upper limit (upper_limit) to be three standard deviations above the mean. Found no rows with outliers as depicted below.



```
print("Highest allowed",df['time_bp'].mean() + 3*df['time_bp'].std())
print("Lowest allowed",df['time_bp'].mean() - 3*df['time_bp'].std())
```

```
Highest allowed 15.406690710730404
Lowest allowed 1.4792667360781078
```



```
# no outliers
str = 'time_bp'
df[(df[str] > 15.40) | (df[str] < 1.48)]
```

0 rows

Also performed outlier detection for numerical columns using **Z-score** method from Scipy Stats library and shows no outliers.

5. **Data Imbalances Check**
   As need to perform classification on gender columns and found no imbalances on gender and both contributing almost equally and therefore no need to treat this data.

6. **Principle Component Analysis (PCA)**
   Decided to refrain from performing Principal Component Analysis (PCA) seeing there is limited number of columns in the original datasets

7. **One Hot Encoding and Label encoding**
   The respondent data was processed using the Scikit-learn library in Python. Categorical features were One-Hot encoded to convert them into numerical format. For the classification task related to gender, Label Encoding was applied since the values are ordinal. The dataset was then divided into training and testing sets for model training. To minimize the influence of outliers, Standard Scaling was performed on the data.

8. **Standardization via StandardScaler**
   To reduce the effect outliers and improve the accuracy of predicative models Standard scaling is performed using scikit preprocessing module for classification and regression and therefore found accuracy to be improving.

VII. Machine Learning Based Predictive Models

1. Perform classification

Predicting gender, whether male or female, is a binary classification problem that necessitates the use of supervised learning models. The primary goal of implementing these models is to accurately identify the gender of respondents based on the features provided. For this purpose, a variety of classifier models were employed, including Logistic Regression, XGBoost, Support Vector Classifier (SVC), Random Forest Classifier, Decision Tree Classifier, Passive Aggressive Classifier, K-Nearest Neighbors and Gradient Boosting Classifier, all implemented using the Scikit-learn library.

The dataset was strategically split into training and testing sets with a 75:25 ratio to ensure the model's ability to generalize to unseen data. Among the models tested, the Support Vector Classifier (SVC) achieved the highest accuracy, with an impressive score of 93.15%. In contrast, the XGBClassifier yielded the lowest performance, with an accuracy of 78.76%. These results highlight the effectiveness of SVC in handling this binary classification problem, while also providing a comparison of various algorithms' performance in gender prediction.

Classifier Model	**Accuracy (%)**	**F1 Score**	**Recall**	**Precision**	**Jaccard Score**	**Log Loss**
LogisticRegression	85.95	0.88	0.91	0.84	0.75	5.06
XGBClassifier	78.76	0.81	0.82	0.80	0.65	7.65
Support Vector Machine	***93.15***	***0.94***	***0.99***	***0.89***	***0.87***	***2.46***
RandomForestClassifier	91.78	0.93	0.95	0.90	0.84	2.96
DecisionTreeClassifier	84.24	0.85	0.85	0.86	0.73	5.68
PassiveAggressiveClassifier	79.11	0.82	0.89	0.77	0.65	7.53
AdaBoostClassifier	91.44	0.92	0.95	0.90	0.84	3.08
KNeighborsClassifier	91.78	0.92	0.91	0.93	0.84	2.96
GradientBoostingClassifier	90.75	0.93	0.92	0.90	0.83	3.33

Table. *Performance evaluation of the Classification models using the various metrices*

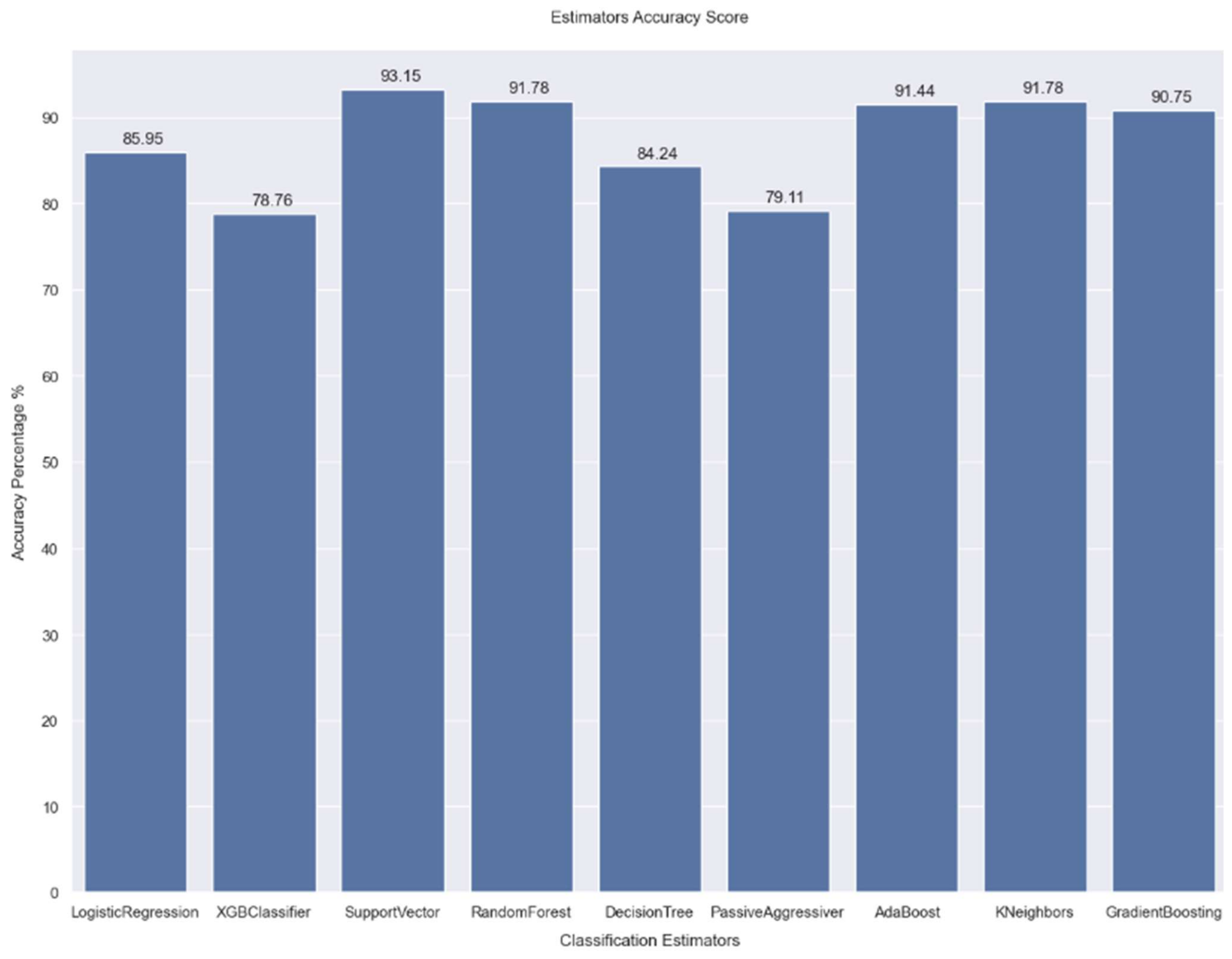


Figure. Performance evaluation of the models using the accuracy score

Best Performing Models: Support Vector, Random Forest, KNeighbors and AdaBoost, as they show high accuracy, strong F1 scores, balanced precision and recall, and low log loss.

Worst Performing Model: XGBClassifer, as it has the lowest accuracy, F1 score, and the highest log loss, indicating poor and less confident predictions.

Balanced Performance: GradientBoostingClassifier show good accuracy, balanced F1, precision, and recall, and reasonably low log loss, making them strong contenders.

In addition to the metrics mentioned above, accuracy_score and ROC AUC score were also calculated to evaluate the performance of the classification models, particularly in the context of binary classification.

To provide better visualization and insight into model performance, the confusion matrix and ROC curve were plotted. These visual tools help to clearly illustrate the effectiveness of the models in distinguishing between the two classes and in identifying any potential areas for improvement.

Hyperparameter tuning was also conducted to identify the optimal parameters for few models, ultimately enhancing the performance of the machine learning models. This tuning process was carried out using both **GridSearchCV on RandomForestClassifier** and **RandomizedSearchCV on XGBClassifier**. These methods systematically search through a range of parameter values, with GridSearchCV exhaustively exploring all possible combinations, while RandomizedSearchCV samples a subset of parameter combinations, providing a more efficient search. By fine-tuning the models through these techniques, able to significantly improve their accuracy and overall performance.

It has been found that GridSearchCV provides better accuracy but at a higher computational cost, while RandomizedSearchCV offers speed but with potentially less accuracy. provides better accuracy but at a higher computational cost, while RandomizedSearchCV offers speed but with potentially less accuracy..This is due to the fact GridSearchCV is an exhaustive search method for hyperparameter tuning. It evaluates all possible combinations of the specified hyperparameters in a grid. Instead of testing all possible combinations, RandomizedSearchCV randomly samples a fixed number of hyperparameter combinations from the specified ranges.

A common approach should be to start with RandomizedSearchCV to quickly identify a promising region in the hyperparameter space. Then, once you have a good understanding of which parameters are likely to be effective, you can use GridSearchCV for a more exhaustive search within that region to fine-tune the model.

There is other tuning technique like **Hyperband** that can be tried which claims to be more efficient hyperparameter optimization technique designed to address some of the limitations of GridSearchCV and RandomizedSearchCV, particularly regarding computational cost and time efficiency. It is especially useful when dealing with large hyperparameter spaces and expensive models.

2. Perform regression

To perform machine learning regression on the " self_time" as target feature, the process begins by selecting an appropriate regression model that best fits the nature of the data. The feature attributes ie independent variables are used to predict the dependent variable or target outcome ie self_time. The regression model is trained on a dataset where the relationship between the feature attributes and the target variable is learned. Various regression techniques, such as Linear Regression, Decision Tree Regression, or Random Forest Regression, can be applied depending on the complexity and distribution of the data.

Once the model is trained, it can predict the target variable for new, unseen data based on the independent variables.
The model's performance is evaluated using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (coefficient of determination) regression score function to assess its accuracy and generalizability. If necessary, hyperparameter tuning can also be employed to optimize the model further.

The various regression models being used can be understood from below figure.

```
if(model_id is 'dummy'):  model = DummyRegressor()
if(model_id is 'lr'):         model = LinearRegression()
if(model_id is 'br'):         model = BayesianRidge()
if(model_id is 'rf'):         model = RandomForestRegressor(n_estimators=10,random_state=10)
if(model_id is 'gbr'):        model = GradientBoostingRegressor(random_state=0)
if(model_id is 'xgb'):        model = XGBRegressor(eval_metric='rmsle')
if(model_id is 'dtr'):        model = DecisionTreeRegressor(random_state=0)
```

Regression Model	**Mean (cv_score)**	**Standard Deviation (cv_score)**	**Mean Squared Error(MSE)**	**Mean Absolute Error(MAE)**	**r2_score**
DummyRegressor	0.54	0.04	0.55	0.44	-0.004
LinearRegression	0.28	0.08	0.26	0.18	0.774
BayesianRidge	0.28	0.08	0.26	0.18	0.775
RandomForestRegressor	0.17	0.17	0.21	0.07	0.856
GradientBoostingRegressor	0.22	0.13	0.22	0.11	0.841
XGBRegressor	0.16	0.19	0.22	0.08	0.833
DecisionTreeRegressor	0.19	0.24	0.27	0.10	0.753

Table. *Performance evaluation of the regression models using the given dataset*

Regression Model Metrics Explanation:

DummyRegressor: This is typically a baseline model that makes predictions based on simple rules (like the mean of the target variable). The R2 score is negative, which means that the model is predicting worse than the mean of the target values but since it's a dummy model, it’s used for comparison rather than practical predictions.

LinearRegression and *BayesianRidge*: These two models have similar performance with a mean cv_score of 0.28, an MSE of 0.26, and an R2 score of 0.77. They have a lower variance (Standard Deviation: 0.08) and are reliable but less accurate than other complex models

RandomForestRegressor: This model has a better mean cv_score (0.17) and a lower MSE (0.21) compared to Linear Regression, with a higher R2 score of 0.85, suggesting better overall performance.

GradientBoostingRegressor and *XGBRegressor*: Both models perform similarly, with a mean cv_score around 0.16-0.22 and an R2 score around 0.84. These models typically outperform simpler models and are good at handling complex relationships in the data.

DecisionTreeRegressor: This model has the MSE (0.22) and the high R2R score (0.753), indicating that it’s fitting to the training data. The high standard deviation (0.24) compared to other model suggests somewhat instability across folds.

In short, it is concluded that:

Best Performing Models: RandomForestRegressor, GradientBoostingRegressor, and XGBRegressor, as they have the lowest MSE, lowest MAE, and high R2 scores.

Worst Performing Model: DecisionTreeRegressor, due to its high MSE, low R2, and instability.

Baseline Comparison: The DummyRegressor is used as a baseline, and most models outperform it, indicating they are adding value over simple baseline predictions.

3. Perform Clustering

K-Means Clustering: Its objective is to divide a set of n observations into k clusters, with each observation assigned to the cluster whose mean (cluster centre or centroid) is closest, thereby acting as a representative of that cluster.

The algorithm aims to minimize the within-cluster sum of squares (WCSS), also known as inertia, by iteratively assigning data points to the nearest cluster centroid and updating the centroids.
The elbow method is employed to find the optimal number of clusters and in this case it came out to be 3 as shown in the graph.

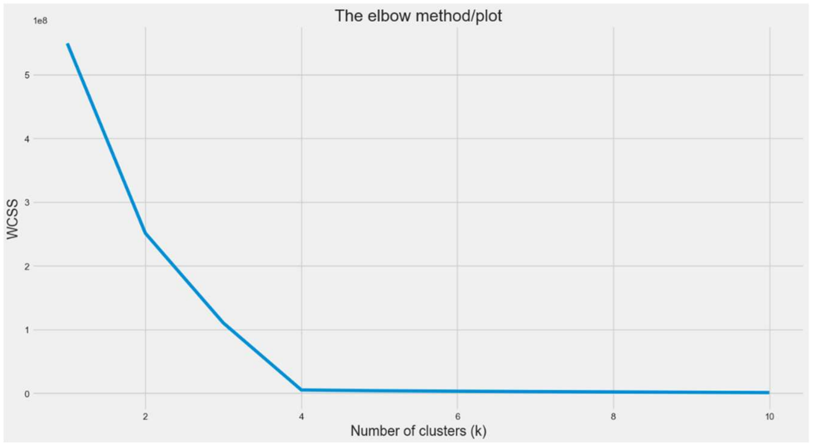


Figure: *Shows the Elbow Points choosing the appropriate number of clusters*

Identifying the elbow point on the SSE curve can be challenging. If you're struggling to pinpoint the elbow, consider using the Python package kneed to determine the elbow point programmatically as shown.

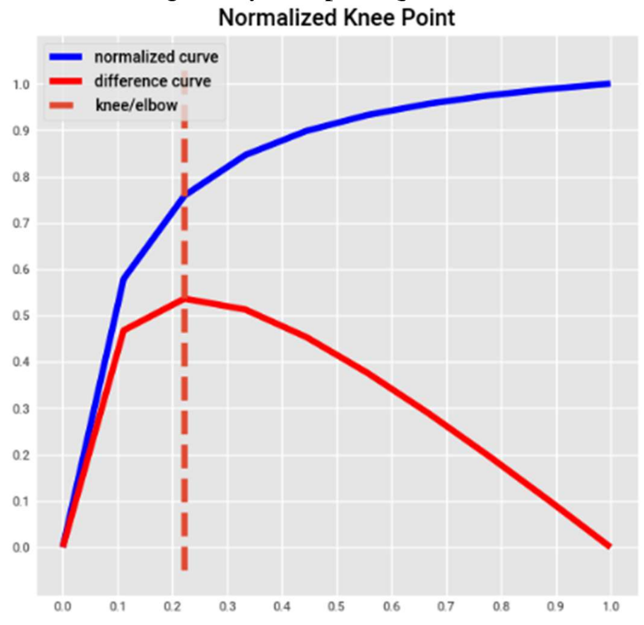
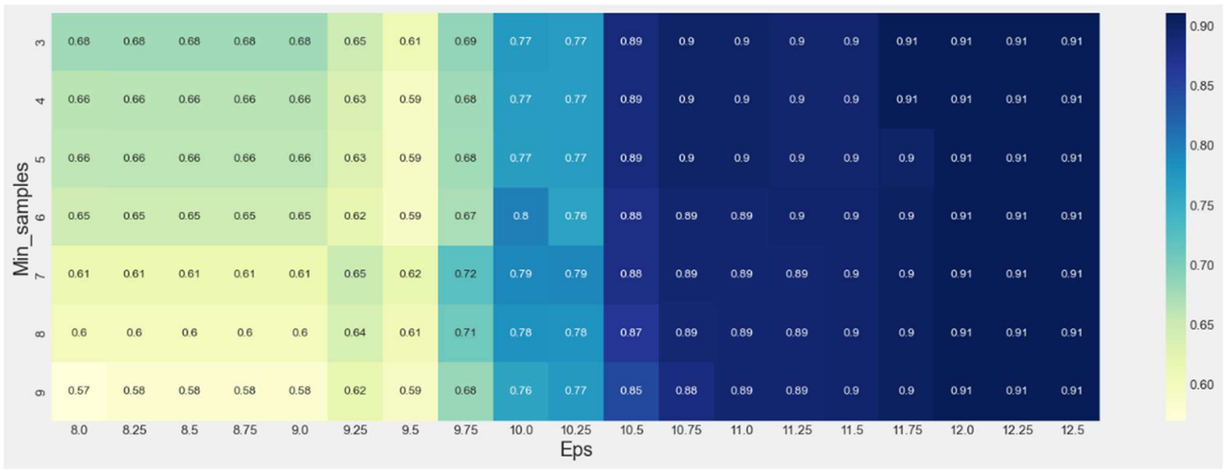


Figure: *Normalized Knee Point plotted using kneed package*

DBSCAN Clustering: It is Density-Based Spatial Clustering of Applications with Noise and makes the clusters based on the parameters like epsilon, min points and noise.
For illustration purpose, DBSCAN model is initialized with eps=0.3, min_samples=20) and tried fitting the model on dataset. The number of clusters is calculated is to be 15 with 647 as number of noise points and Silhouette Coefficient of -0.076 which suggests that the clustering may be less accurate, with overlapping clusters or points that are not well-assigned to their respective clusters.

A large number of clusters might indicate that the eps parameter is too small, causing DBSCAN to create many small, dense clusters instead of a few larger ones. The large number of noise points suggests that many data points don't fit well into any cluster. This can occur if the eps is too small or min_samples is too large, making it difficult for DBSCAN to find sufficiently dense regions. The negative Silhouette Score indicates that the clustering structure might not be suitable, possibly due to inappropriate parameter settings or the intrinsic structure of the data being incompatible with DBSCAN's assumptions.

In order to improve, eps and min_samples have been tried with different values but it did not improve considerably.
eps_values = np.arange(8,12.75,0.25) # eps values to be investigated
min_samples = np.arange(3,10) # min_samples values to be investigated
Then a heatmap is plotted that shows how many clusters are being generated by the DBSCAN algorithm for the respective parameters combinations that shows the number of clusters vary from 48 to 16 but for most of the combinations gives 16-17 clusters.

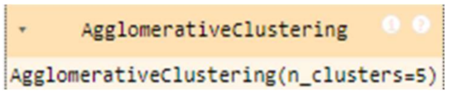


```
Global maximum is 0.91 for eps=11.75 and min_samples=4.
```

Figure: *To decide which combination to choose, a metric - a Silhouette score is being used to plot a heatmap*

Consequently, other clustering algorithms like K-means, Agglomerative Clustering (hierarchical clustering) and Mean-Shift Clustering have been tried and added the results in below table which performed better.

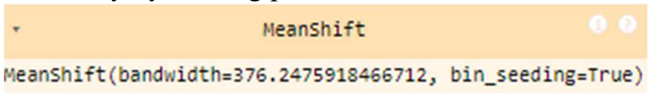
Agglomerative Hierarchical Clustering: Recursively merges pair of clusters of sample data; uses linkage distance.



```
AgglomerativeClustering
AgglomerativeClustering(n_clusters=5)
```

Here we initialized the model with n_clusters=5 and fitted and predicted on the model and got the Silhouette score of 0.769 which is quite good and indicates that the data points are well-clustered, with clear separation between clusters and tight cohesion within each cluster.

Mean-Shift Clustering: Which works by that assigning the datapoints to the clusters\ iteratively by shifting points towards the mode.



```
MeanShift
MeanShift(bandwidth=376.2475918466712, bin_seeding=True)
```

Here we initialized the model with bandwith and bin_seeding over sample of 1000, fitted and predicted on the model and got the Silhouette score of 0.917 which is quite good and indicates that the data points are well- clustered, with clear separation between clusters and tight cohesion within each cluster.

Comparative Analysis of Clustering Algorithms

Clustering Algorithms	**Clustering Methods**	**silhouette_score**	**davies_bouldin_score**
KMeans Clustering	Partitioning methods	0.917	0.130
DBSCAN	Density-based Clustering	-0.076	1.049
Agglomerative Clustering	Hierarchical Clustering	0.769	0.438
Mean-Shift Clustering	Density-based Clustering	0.917	0.131

Table: *The table presents a comparative analysis of different clustering algorithms based on their clustering methods, silhouette scores, and Davies-Bouldin Index (DBI) scores.*

- *KMeans Clustering* and *Mean-Shift Clustering* both belong to the partitioning methods category. They show the highest silhouette score of **0.917**, indicating good clustering performance. KMeans has a slightly lower Davies-Bouldin score of **0.130**, suggesting better-defined clusters compared to Mean-Shift's **0.131**.
- *DBSCAN*, a density-based clustering algorithm, has the lowest silhouette score at **-0.076**, indicating poor clustering quality. However, its Davies-Bouldin score is **1.049**, higher than other models, implying that while clusters might be less distinct, they are somewhat compact.
- *Agglomerative Clustering*, a hierarchical clustering method, has a silhouette score of **0.769**, which is decent but lower than KMeans and Mean-Shift. Its Davies-Bouldin score is **0.438**, consistent with most other algorithms, reflecting average cluster separation and compactness.

In summary, KMeans Clustering stands out as the most effective algorithm based on the metrics provided, while DBSCAN exhibits the weakest performance. Mean-Shift Clustering also behaved almost same as KMeans.

VIII. DATASET SHIFT - IMPACT ON MODEL PERFORMACE

Given the nature of the COVID-19 pandemic and the rapidly changing environment, both ***label and covariate shifts*** are highly likely on workforce adaptation. As remote work preferences and demographics evolve, these shifts could degrade your model's performance over time.

Imagine the mentioned models was trained to predict preferences for remote work based on data collected during the early months of the COVID-19 pandemic, where a significant number of respondents preferred remote work due to health concerns. However, as time passes and vaccines become widely available, more people might prefer returning to in-person work. This change in preference alters the distribution of the target variable (label), even though the features (age, gender, occupation, etc.) remain the same.

Other the other hand, covariate shift could happen if, over time, the demographic profile of the workforce shifts—perhaps younger professionals become more prevalent in remote work settings, or certain job types become more dominant. Even if the relationship between these features and the preference for remote work (the label) stays constant, the shift in feature distribution can still confuse the model.

To conclude, it is essential to:

-> Implement ongoing monitoring mechanisms to detect shifts in the distribution of both features and labels. This could involve statistical tests, visual inspections like KDE plots, or tracking model performance metrics for sudden changes

-> Regularly update the models with the latest data to ensure they remain relevant. This could involve re-training, fine-tuning, or employing adaptive learning techniques that allow the model to evolve with the data.

-> Use robust evaluation strategies, such as cross-validation with temporal splits, to account for potential shifts and ensure the model is generalizing well to the current data distribution

IX. EXPLAINABLE AI (XAI)

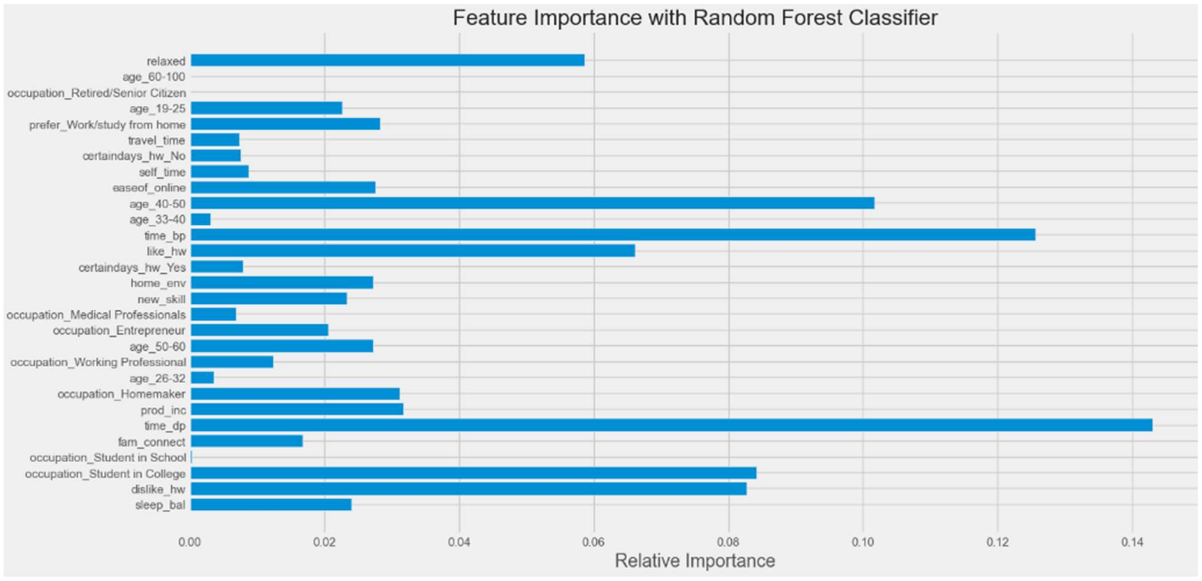
Explainable AI (XAI) is essential for understanding predictions related to workforce behaviour and preferences, ensuring that decisions based on these models are both reliable and justifiable. Tools like feature importance and coefficients in models such as Random Forest and Decision Tree Classifier help identify which factors, like gender or age, most influence predictions, such as preferences for remote work.

Local interpretation tools like ***LIME*** and ***SHAP*** enhance trust and accountability by making individual predictions more understandable. For example, SHAP values clarify how different factors contribute to predictions in complex models like Random Forest.

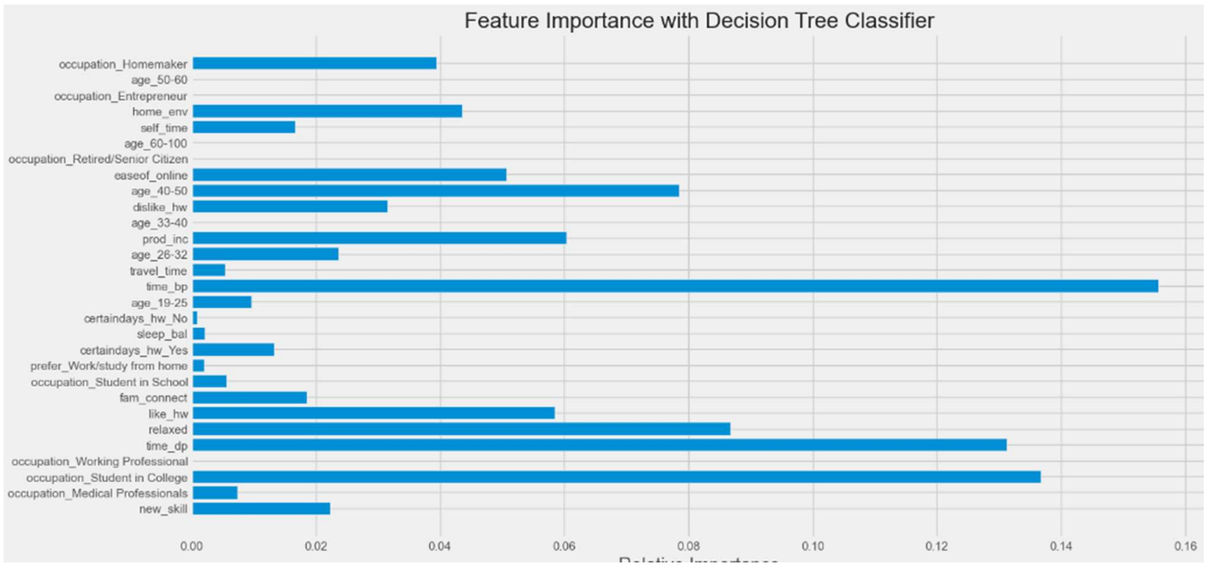
Additionally, explainable methods aid in interpreting clusters by analysing centroids and silhouette scores, providing insights into workforce segmentation. These techniques are critical for uncovering the most influential features in classification models, enabling a deeper understanding of how input variables impact predictions and ultimately fostering trust in machine learning systems.

In this context, let's explore the different ***feature importance*** techniques used in classification models:

a. Random Forest Feature Importance



b. Decision Tree Feature Importance



X. CONCLUSION

This study investigates the factors influencing an individual's mindset in a remote setting by leveraging data exploration techniques and predictive modelling through machine learning. Various machine learning classification models were applied with *gender* as the target variable, while regression models were employed to predict outcomes based on the *self_time* feature. Additionally, multiple clustering models were analysed, accompanied by relevant metrics and visualizations. A comprehensive comparative analysis of all the mentioned algorithms was also conducted.

XI. REFERENCES

Matloff, N. Statistical Regression and Classification: From Linear Models to Machine Learning; Chapman and Hall/CRC: Boca Raton, FL, USA, 2017

Olivas, E.S. Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques: Algorithms, Methods, and Techniques; IGI Global: Hershey, PA, USA, 2009.

Webb G.I. (2011) Naïve Bayes. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_576

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011

Agglomerative Clustering, https://scikitlearn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

Correlation, https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/correlation-pearson-kendall-spearman/

Sweetviz Report, https://pypi.org/project/sweetviz/

Pandas corr, https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.corr.html

XII. KEY TERMS AND DEFINITIONS

Accuracy - The number of correct predictions divided by the total number of predictions.
F1 Score - harmonic mean of the recall and precision.
Recall - number of true positives divided by the sum of the number of true positives and number of false negatives.
Precision - number of true positives divided by sum of the number of true positives and the number of false positives.
Jaccard score (or Jaccard index) measures similarity between the predicted and actual classes. A higher Jaccard score indicates better performance,
*Log loss***,** or logistic loss, measures the uncertainty of the model's predictions. It’s particularly useful for evaluating probabilistic classification models. Lower log loss indicates better performance
CV Score is the average score of the model across multiple cross-validation (cv) folds. A lower mean cv_score generally indicates better performance because it implies that the model makes predictions closer to the actual values
Davies-Bouldin Index (DBI) is a metric used to evaluate the performance of a clustering algorithm. It measures the average similarity ratio of each cluster with its most similar cluster, where similarity is defined as the ratio of within-cluster distances to between-cluster distances