DATA COLLECTION PIPELINE FOR XYZ COMPANY

Done by: Abdulrahman Afifi

Team Member's details

Group Name: The Potent self

Name: Abdulrahman Afifi

Email: abdulrahmanafifi33@gmail.com

Country: Turkey

College/Company: Bahcesehir University

Specialization: Data Analyst

AGENDA

Problem Description

Data Cleansing and Transformation

Visual Aids for EDA

Recommended models for this data set

Summary

PROBLEM DESCRIPTION

XYZ firm is gathering client data using Google Forms/Survey Monkey and has published a variety of n forms on the web.
The company want to build a pipeline that would collect all of the data from these Google forms/survey surveys and visualise it on the dashboard.

The company needs clean data, and if there are any data issues in the data, they should be addressed via this process (duplicate data or junk data). A dedup check should be run on the customer's email address.

GITHUB REPO LINK

DataCollectionPipelineProject/Week11 at main · AfifiGhost2000/DataCollectionPipelineProject (github.com)

DATA CLEANSING &TRANSFORMATION

After collecting our data from google forms using specific API, we stored it into a pandas dataframe object to apply some data cleansing and transformation. The first step in this procedure is that I renamed and shortened long questions available in the form to make it easier for data analysis. Moreover, I added the +sign missing from phone numbers provided by the customers. I also extracted day,month,hour,min, and seconds from the timestamps available to be able to analyse and forecast our data.
Email Id is being extracted from the email to avoid duplicated emails from same user. Also machine learning could be applied to emailId to identify username.

The dollar sign has been deleted from the average monthly income to be able to analyse our data properly.

Also Satisfaction rate has been segregated into 3 categories 'Low', 'Medium', and 'High'. This can provide some useful insights for the company.

FINAL REPRESENTATION OF OUR DATA

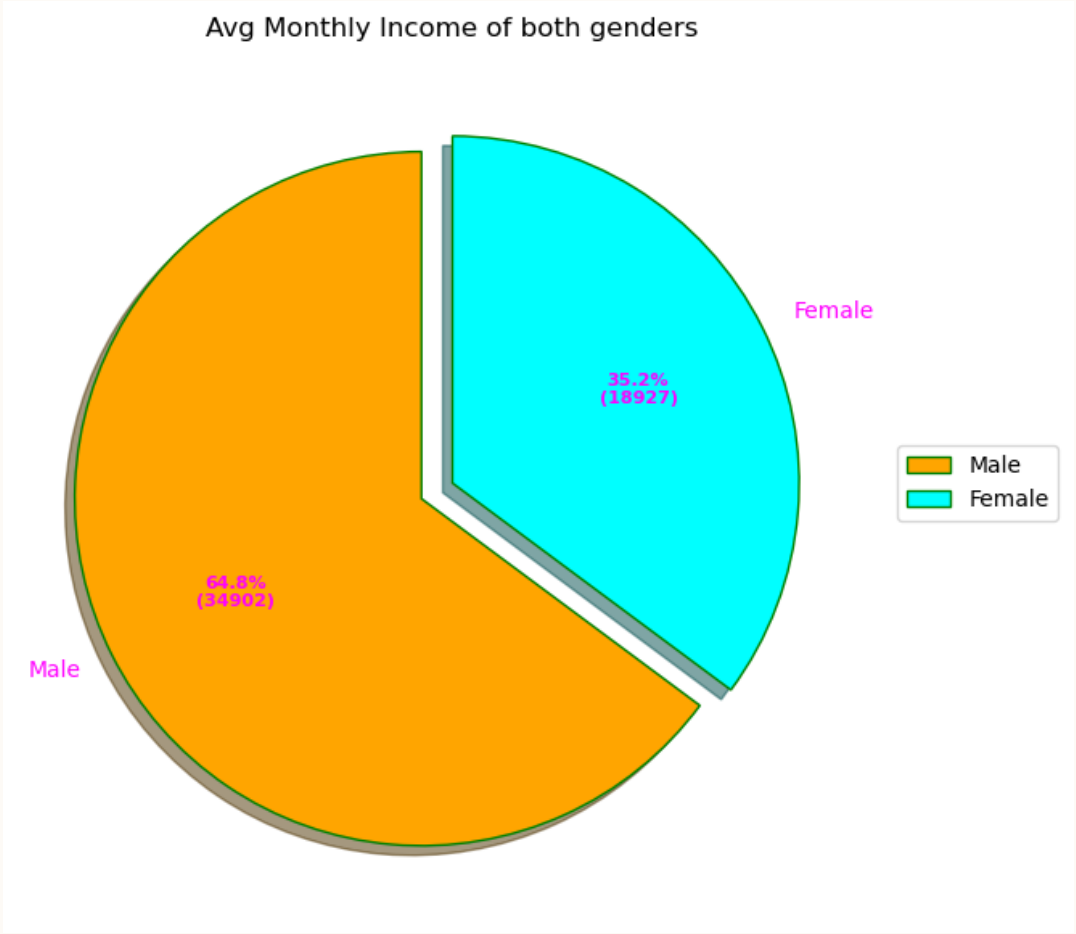
	Timestamp	month	day	hour	minute	second	Name	Country	Age	Email	Address	Phone number	Gender	Satisfaction Rate	Avg Monthly Income
0	2023-01-30 17:57:49	1	30	17	57	49	Abdulrahman Afifi	Egypt	22	abdulrahmanafifi33@gmail.com	Nardenk Sokağı, Abbasağa Mahallesi 34022, Ista...	+905528499159	Male	4	25000.0
1	2023-01-30 18:05:17	1	30	18	5	17	Alex John	Argentina	34	alex.john35@gmail.com	Ariensplein 1, Enschede, Netherlands	+31684475461	Male	3	34000.0
2	2023-01-30 18:08:13	1	30	18	8	13	Rebecca David	UK	45	rebecca49david@gmail.com	Beşiktaş İstanbul ıhlamurdere caddesi, Aşık Ga...	+415521701586	Female	1	12000.0
3	2023-02-10 17:20:13	2	10	17	20	13	Bill Gates	Turkey	82	jnds@gmail.com	XXXXX	+415597015875	Male	3	58000.0
4	2023-02-10 18:00:13	2	10	18	0	13	Elon Musk	USA	43	emusk@gmail.com	XXYXX	+415597015876	Male	4	28000.0

DATA DESCRIPTION

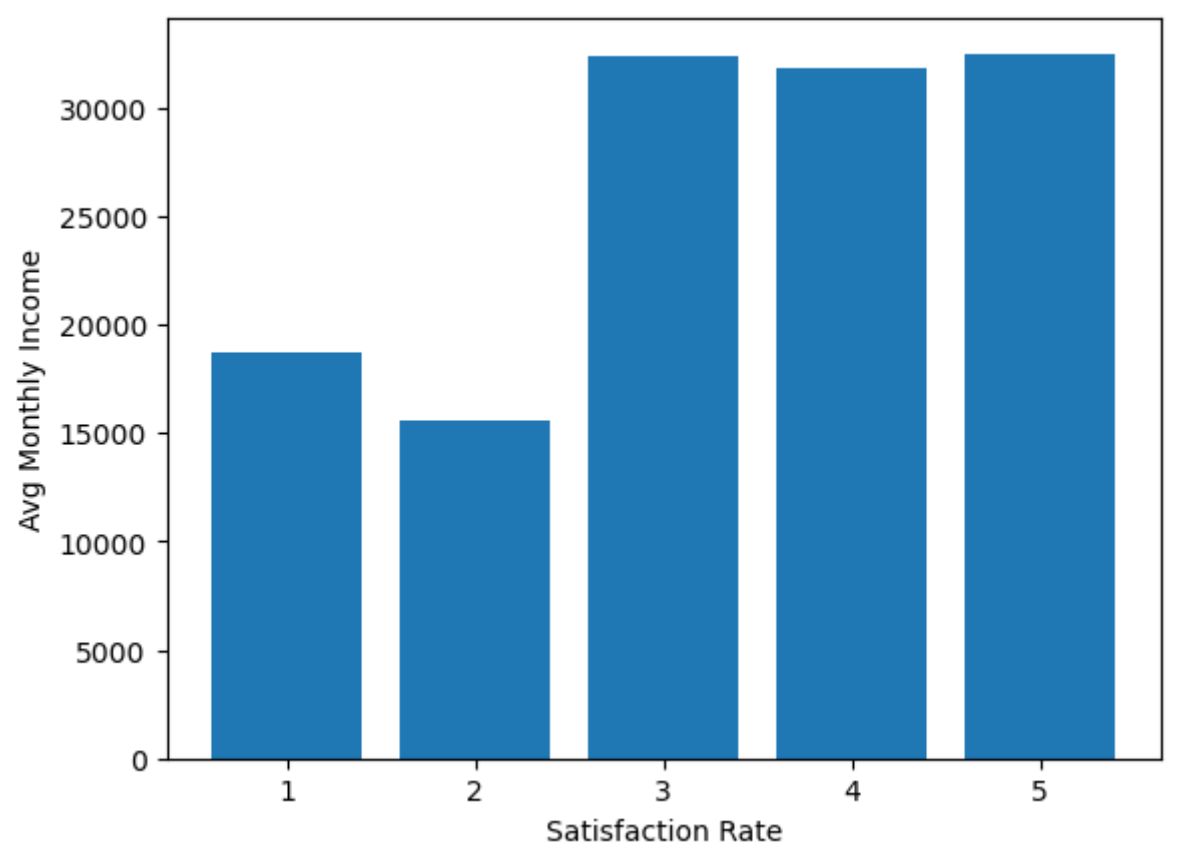
	Low Satisfaction Rate		Medium Satisfaction Rate		High Satisfaction Rate	
	Age	Avg Monthly Income	Age	Avg Monthly Income	Age	Avg Monthly Income
count	5.00	5.00	4.00	4.00	11.00	11.00
mean	34.00	17481.80	42.00	32364.00	47.27	32126.00
std	11.07	5196.52	26.98	18546.43	15.18	12076.54
min	23.00	12000.00	24.00	18456.00	22.00	16496.00
25%	24.00	13212.00	27.00	18864.00	38.50	23500.00
50%	32.00	18000.00	31.00	26500.00	49.00	28714.00
75%	45.00	19197.00	46.00	40000.00	58.00	38267.50
max	46.00	25000.00	82.00	58000.00	67.00	57896.00

VISUAL AIDS FOR EDA

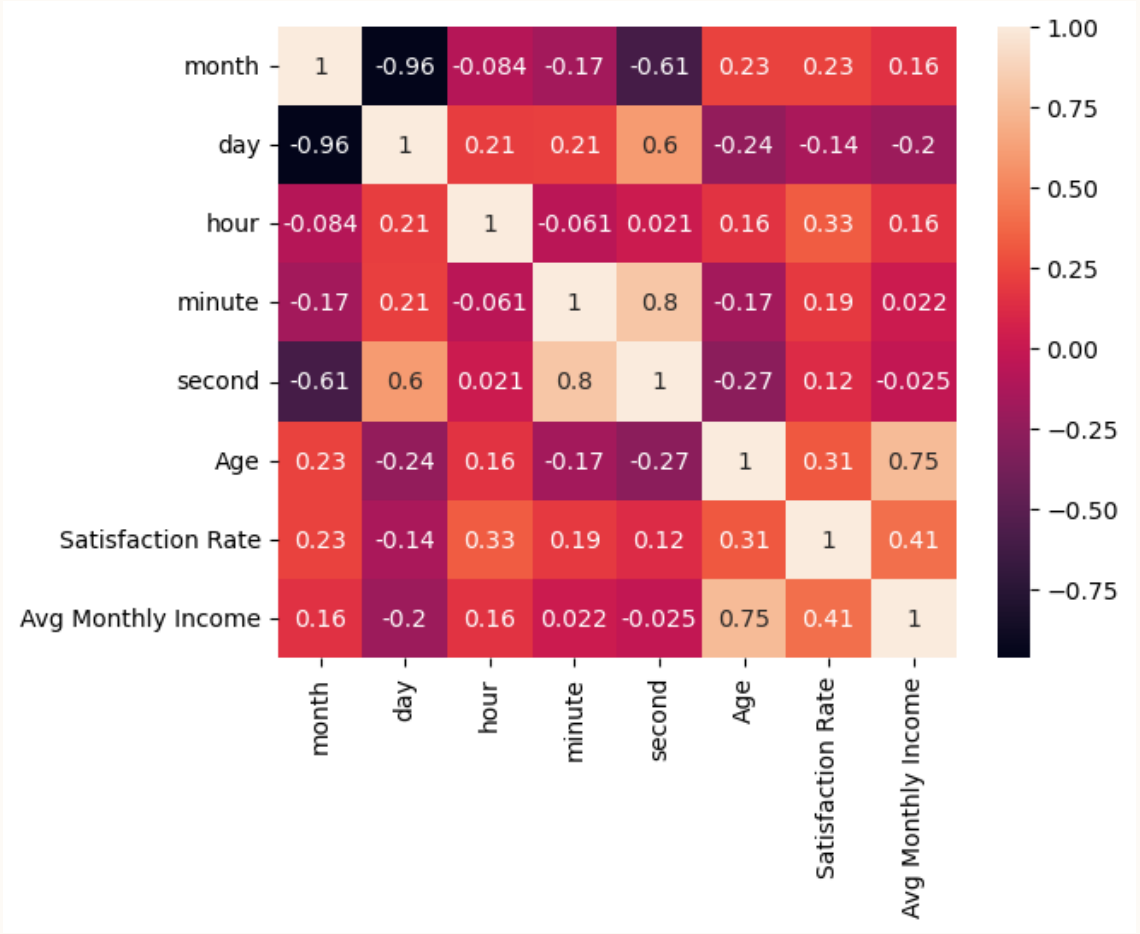
GENDERWISE AVG MONTHLY INCOME PERCENTAGE



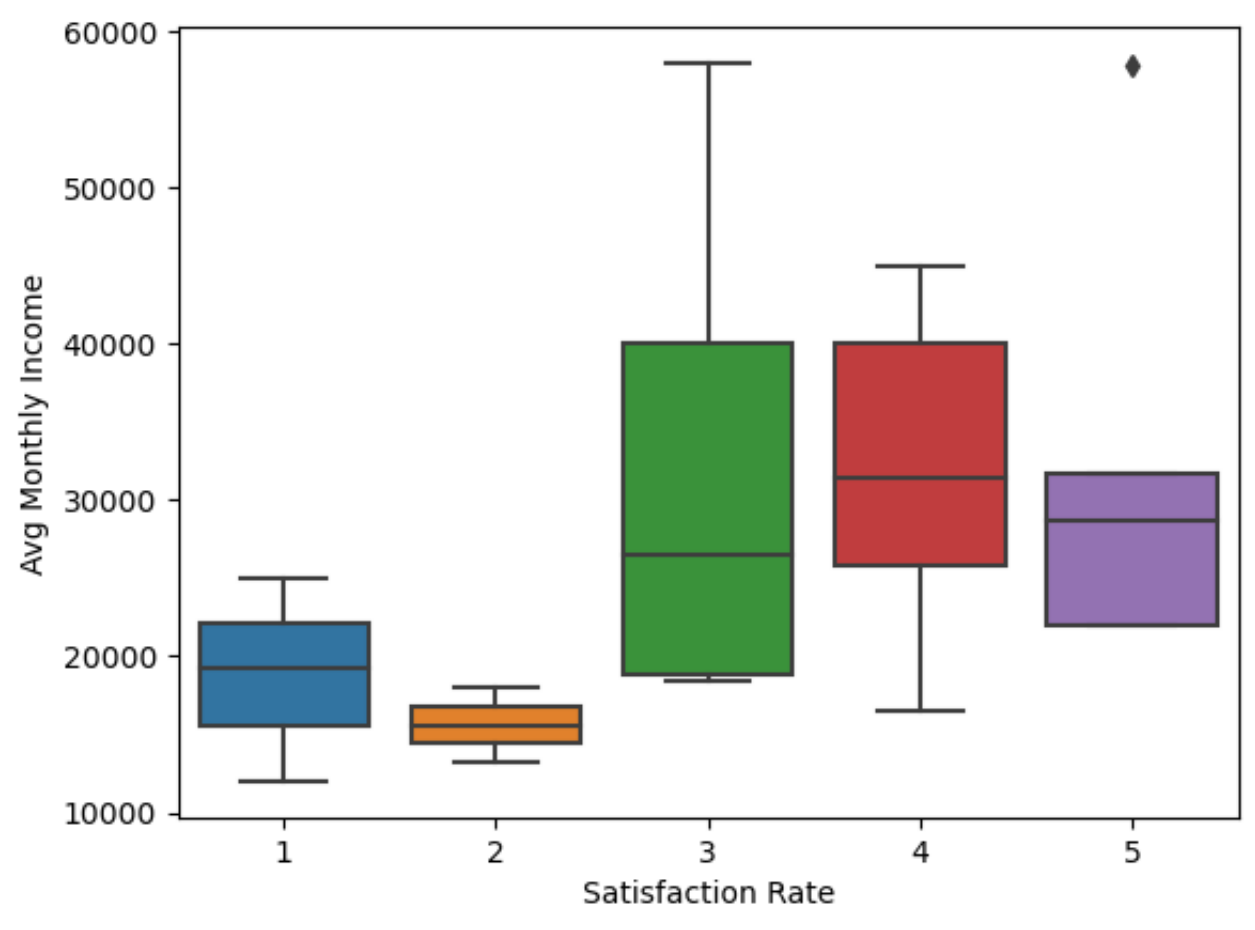
AVG MONTHLY INCOME PER SATISFACTION RATE FOR ALL CUSTOMERS WHO RESPONDED



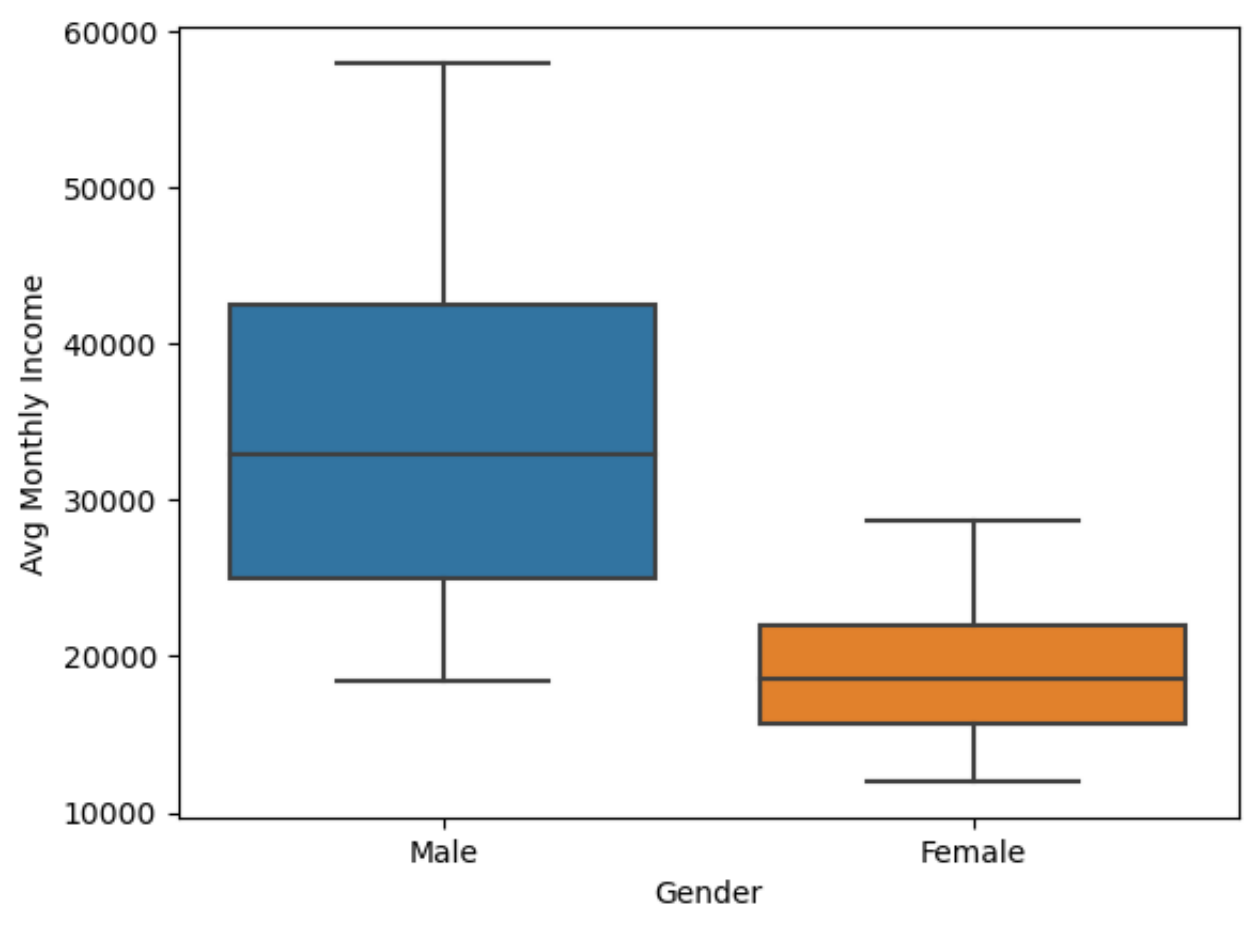
HEATMAP REPRESENTATION



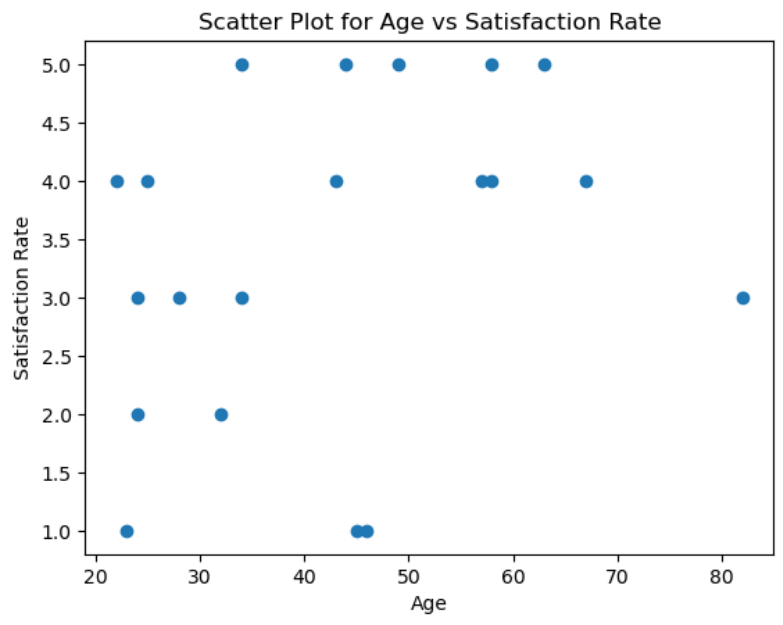
BOX PLOT FOR SATISFACTION RATE VS AVG MONTHLY INCOME

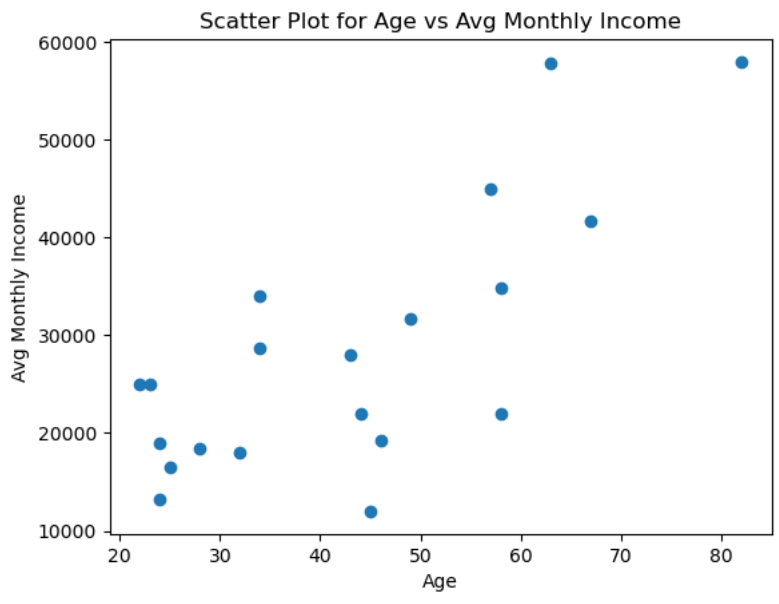


BOX PLOT FOR GENDER VS AVG MONTHLY INCOME

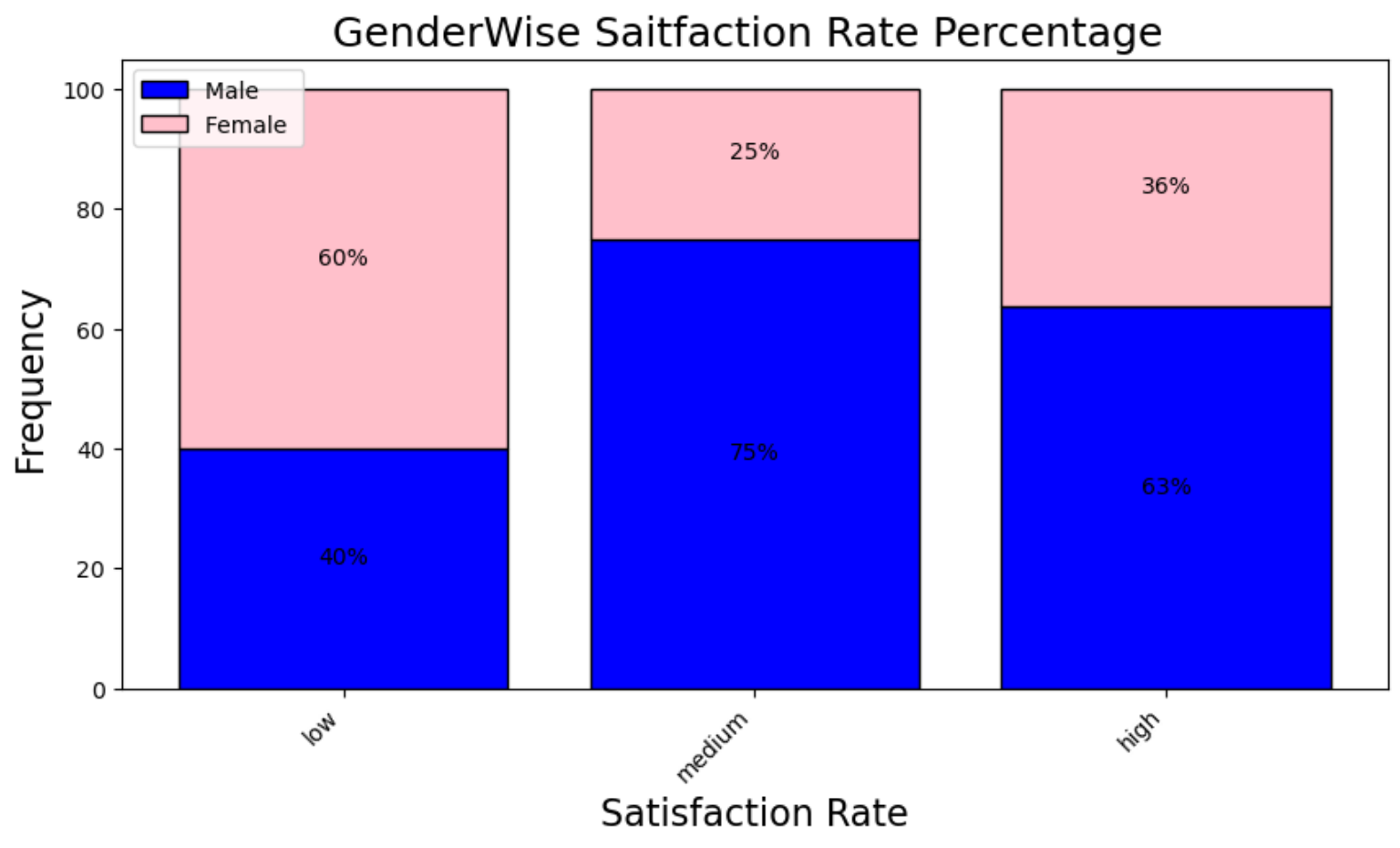


SCATTER PLOT

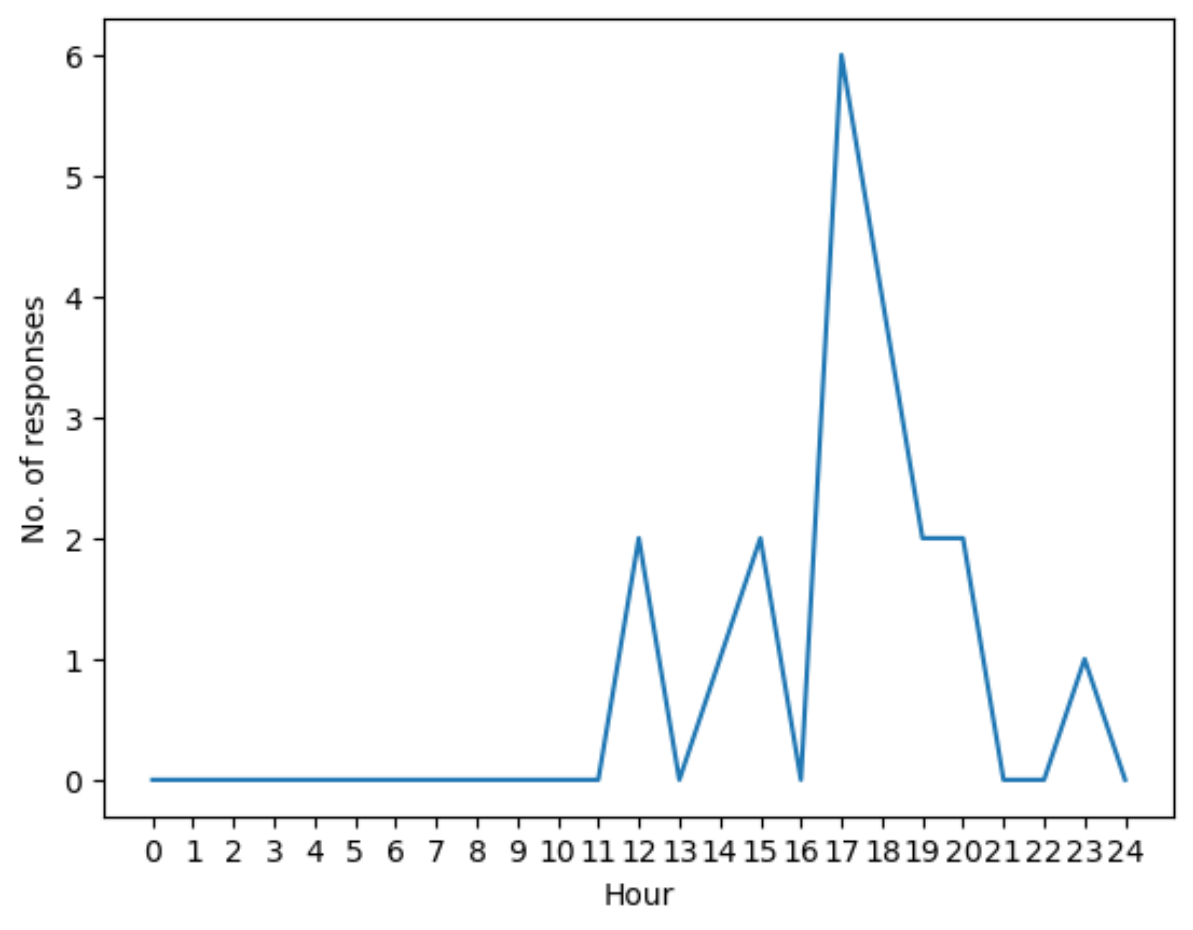




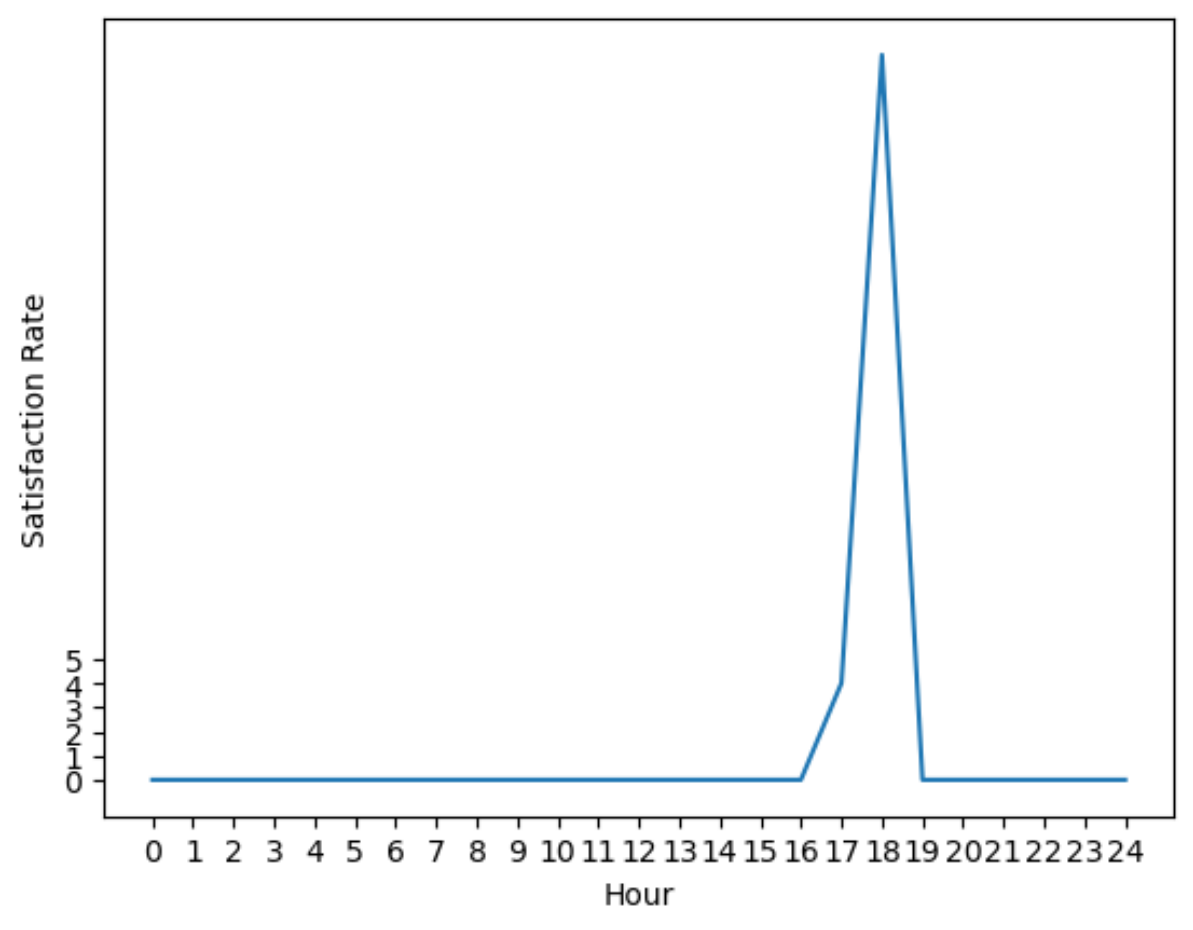
GENDER-WISE SATISFACTION RATE PERCENTAGE



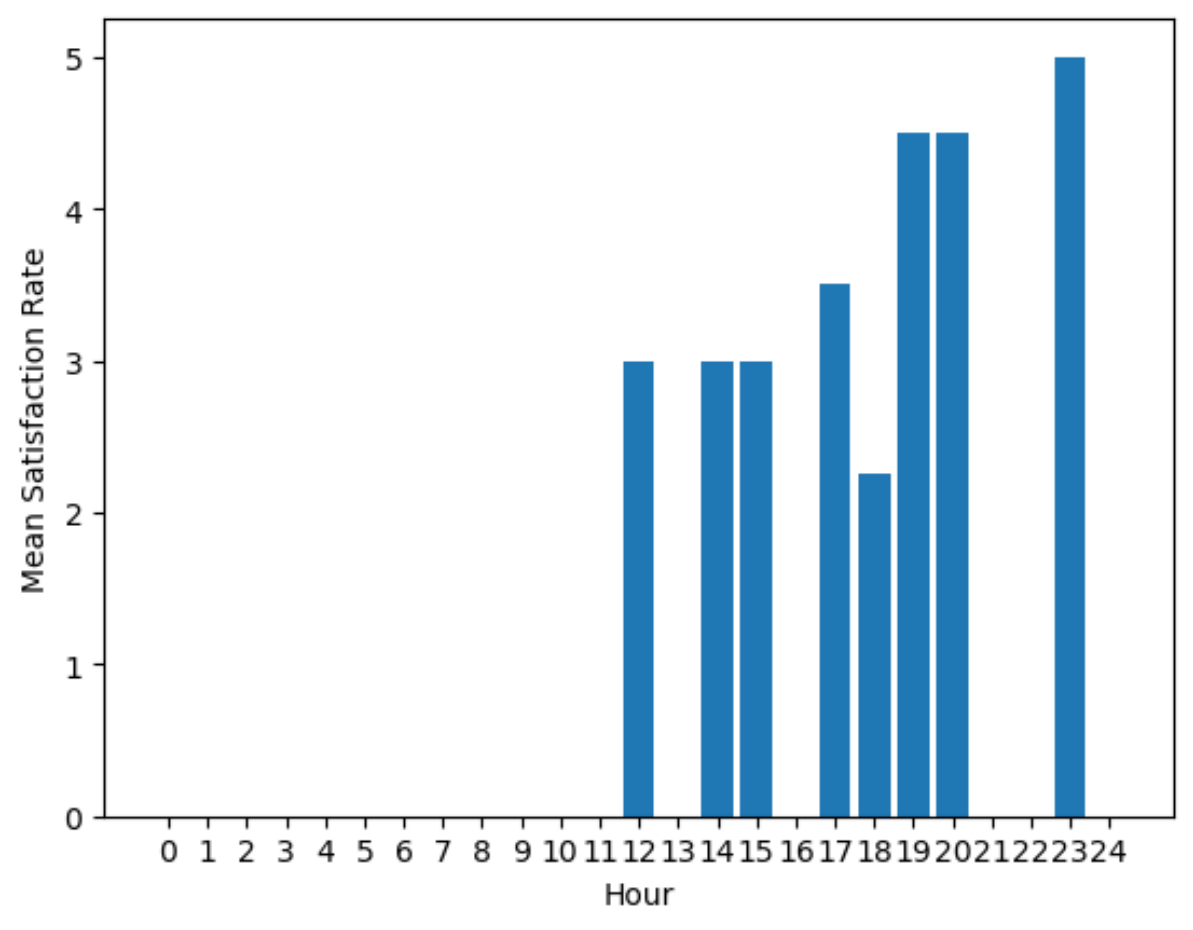
HOURLY NO. OF RESPONSES



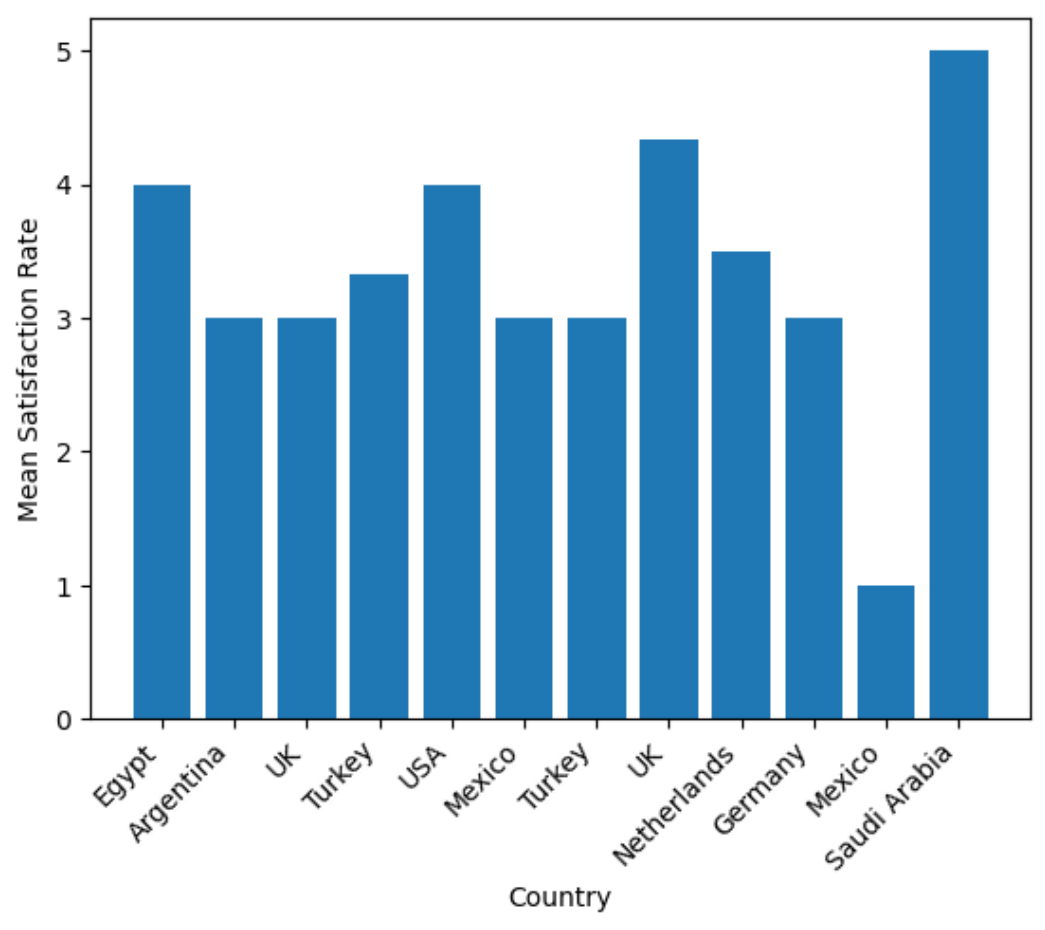
SATISFACTION RATE PER TIME SLOT(IN DECIMALS)



MEAN SATISFACTION RATE PER HOUR

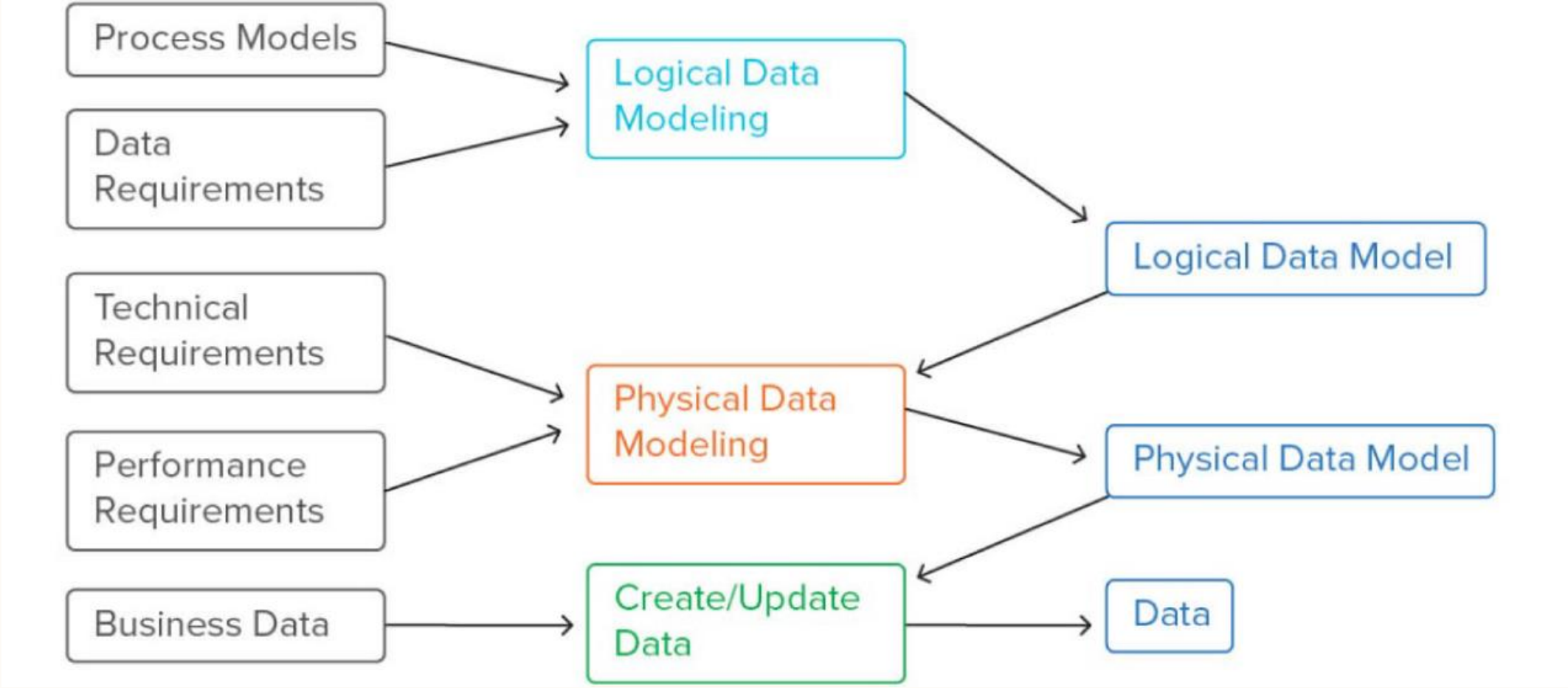


MEAN SATISFACTION RATE PER COUNTRY



RECOMMENDED MODELS FOR THIS DATASET

FUNDAMENTAL UNDERSTANDING



Machine Learning Algorithms *(sample)*

	Unsupervised	Supervised

Continuous	• Clustering & Dimensionality Reduction
	  ○ SVD
	  ○ PCA
	  ○ K-means
		• Regression
		  ○ Linear
		  ○ Polynomial
		• Decision Trees
		• Random Forests

Categorical	• Association Analysis
	  ○ Apriori
	  ○ FP-Growth
	• Hidden Markov Model
		• Classification
		  ○ KNN
		  ○ Trees
		  ○ Logistic Regression
		  ○ Naive-Bayes
		  ○ SVM

BENEFITS

- Data models can help us emphasize on what data is needed and how it can be organised into our database.
- The obvious primary key found in our database is Timestamp field.
- After visualising our data model we must choose a suitable machine learning model for efficient data analysis and model prediction. Because, as the data grows exponentially we need a machine learning model to predict values instantly.
- Some of well know machine learning model techniques are: Supervised/Unsupervised learning and reinforcement learning. Can be classified as linear, ensemble, boosting, and stacking.

FEATURE SELECTION

```
X = df[['Country', 'Gender', 'Age', 'Satisfaction Rate_label']]
y = df['Avg Monthly Income']
```

The primary key can be excluded from our feature set. Then, we begin exploring other features. I decided to choose only Country, Gender, Age, and Satisfaction Rate label as X feature set. Because they are easier to analyse and encode. Avg Monthly Income is being selected as the Y feature set. This is is the value we would like to predict.

MODEL PREPARATION

I encoded categorical or non-float values to be able to fit it in our machine learning models.

```
models=[LogisticRegression(),
LinearSVC(),
SVC(kernel='rbf'),
RandomForestClassifier(),
DecisionTreeClassifier(),
GradientBoostingClassifier(),
GaussianNB()]

model_names=['LogisticRegression','LinearSVM','rbfSVM',
 'RandomForestClassifier', 'DecisionTree',
'GradientBoostingClassifier', 'GaussianNB']

label = LabelEncoder()

df['Satisfaction Rate_label'] = label.fit_transform(df['Satisfaction Rate_label'])

integerMappingRate = get_integer_mapping(label)

label = LabelEncoder()


df['Gender'] = label.fit_transform(df['Gender'])

df['Gender'] = df['Gender'].astype('category')
df['Gender'] = df['Gender'].cat.codes
```

EXPLANATION

Data is split into 30% test and 70% train. Then we fit our data into all possible models and evaluate accuracy. Output of code is below!

```
{'Modelling Algorithm': ['LogisticRegression',
  'LinearSVM',
  'rbfSVM',
  'KNearestNeighbors',
  'RandomForestClassifier',
  'DecisionTree',
  'GradientBoostingClassifier',
  'GaussianNB'],
 'Accuracy': [0.5,
  0.8333333333333334,
  0.6666666666666666,
  0.6666666666666666,
  0.6666666666666666,
  0.6666666666666666,
  0.6666666666666666,
  0.6666666666666666]}
```

```
##Categorical modeltype##
X = df[['Country', 'Gender', 'Age', 'Satisfaction Rate_label']]
y = df['Avg Monthly Income_label']

x_train,x_test,y_train,y_test=train_test_split(X, y ,test_size=0.30,random_state=42)



acc=[]
eval_acc={}
for model in range(len(models)):
    classification_model=models[model]
    classification_model.fit(x_train,y_train)

    sc=StandardScaler()
    x_train=sc.fit_transform(x_train)
    x_test=sc.transform(x_test)

    y_pred=classification_model.predict(x_test)
    acc.append(accuracy_score(y_test,y_pred))
    eval_acc={'Modelling Algorithm':model_names,'Accuracy':acc}
eval_acc
```

SOME USEFUL FUNCTIONS NEEDED

```
def get_integer_mapping(le):
    '''
    Return a dict mapping labels to their integer values
    from an SKlearn LabelEncoder
    le = a fitted SKlearn LabelEncoder
    '''
    res = {}
    for cl in le.classes_:
        res.update({cl:le.transform([cl])[0]})

    return res

integerMapping = []

def get_label_encode(x):
    try:
        my_int = int(x)
        return my_int
    except ValueError:
        return int(integerMapping[x])
```

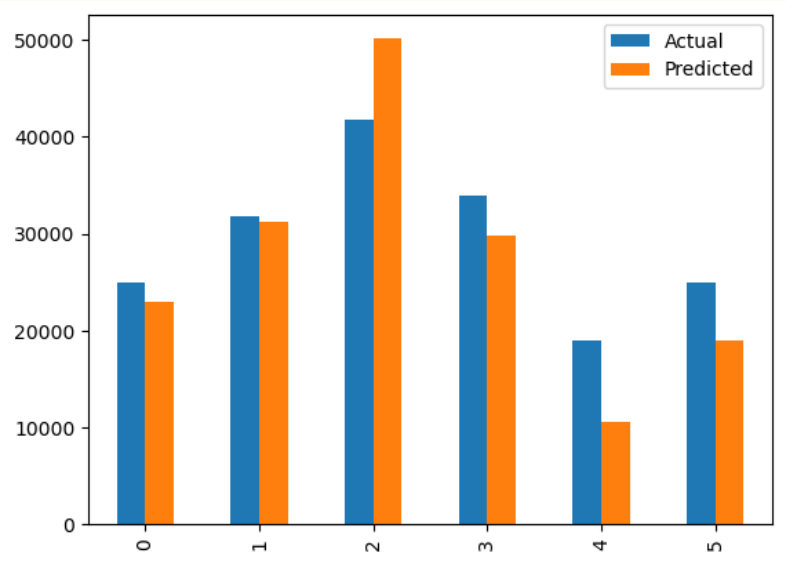
ACTUAL VS PREDICTED VALUES FROM LINEAR REGRESSION MODEL

```
regressor = LinearRegression()
regressor.fit(x_train,y_train)
# predict the y values
y_pred=regressor.predict(x_test)
# a data frame with actual and predicted values of y
evaluate = pd.DataFrame({'Actual': y_test.values.flatten(),
'Predicted': y_pred.flatten()})
evaluate.head(10)
```

✓ 0.3s

	Actual	Predicted
0	25000.0	22955.347885
1	31745.0	31188.525482
2	41746.0	50117.714617
3	34000.0	29862.411101
4	19000.0	10613.533441
5	25000.0	18976.588389

```
evaluate.head(10).plot(kind = 'bar')
```



CALCULATE ACCURACY

```
# Scoring the model
from sklearn.metrics import r2_score, mean_squared_error,mean_absolute_error

# R2 Score
print(f"R2 score: {r2_score(y_test, y_pred)}")
# Mean Absolute Error (MAE)
print(f"MSE score: {mean_absolute_error(y_test, y_pred)}")
# Mean Squared Error (MSE)
print(f"MSE score: {mean_squared_error(y_test, y_pred)}")
```

✓ 0.3s

```
R2 score: 0.3916155523753412
MSE score: 4920.051386423646
MSE score: 33051637.076261606
```

OTHER ML MODELS

- To use other models, we must convert our avg monthy income to categorical values based on some criteria/threshold. Since these models work best in predicting categorical data type.

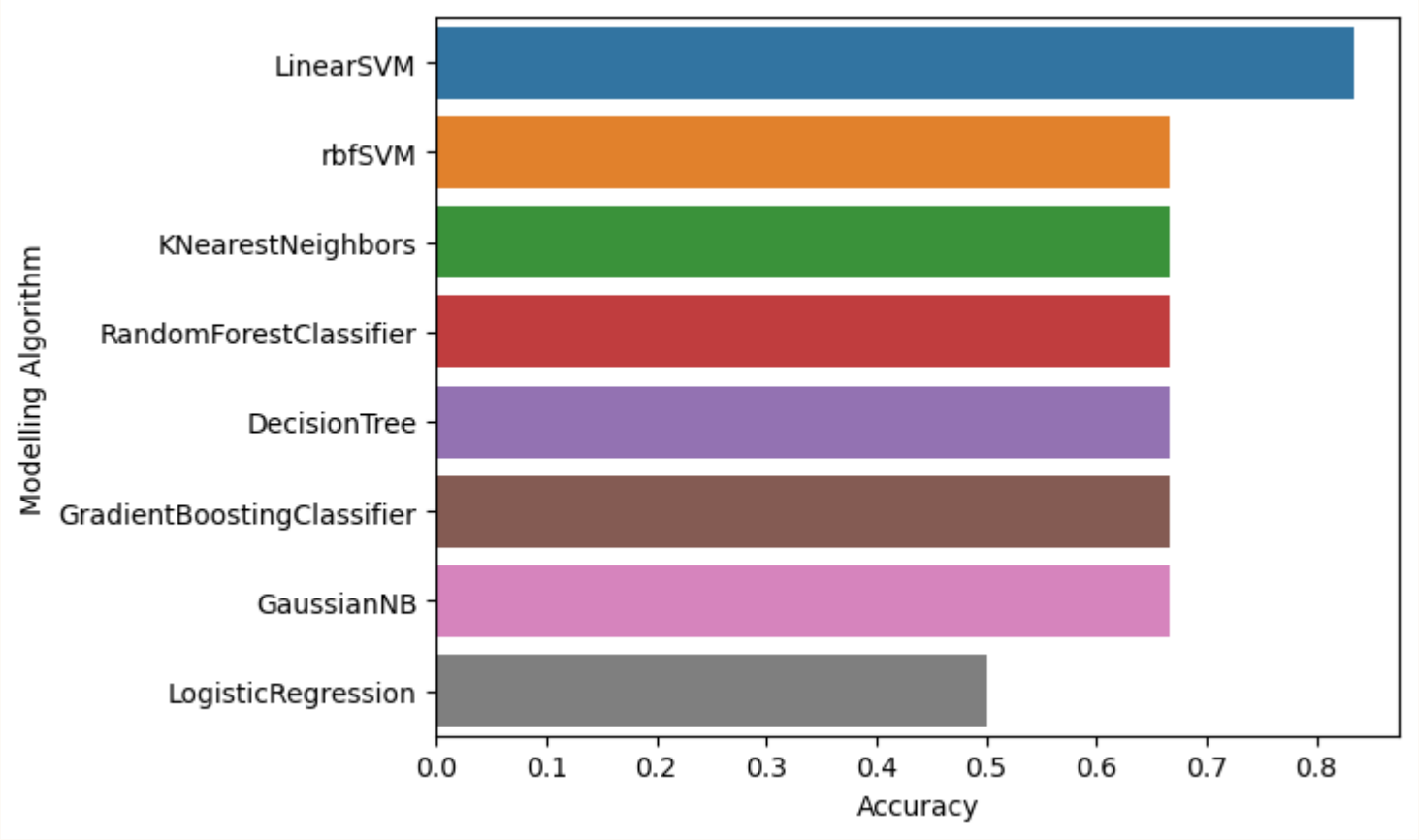
```
df['Avg Monthly Income_label'] = df['Avg Monthly Income'].apply(lambda value: ('low' if value < 13000 else 'medium') if value <= 30000 else 'high')
df['Avg Monthly Income_label'] = pd.Categorical(df['Avg Monthly Income_label'], categories=['low','medium','high'])
```

RESULTS

	Modelling Algorithm	Accuracy
1	LinearSVM	0.833333
2	rbfSVM	0.666667
3	KNearestNeighbors	0.666667
4	RandomForestClassifier	0.666667
5	DecisionTree	0.666667
6	GradientBoostingClassifier	0.666667
7	GaussianNB	0.666667
0	LogisticRegression	0.500000

```
sns.barplot(y='Modelling Algorithm',x='Accuracy',data=acc_table)
```

✓ 0.1s



CONCLUSION

Based on the results, we can choose LinearSVM

Accuracy = 83.3%

THANK YOU

Done by: Abdulrahman Afifi

Don’t forget to visit my:
LinkedIn Profile