HEALTHCARE ATTRITION

SC1015 MINI PROJECT

C133
SHRUTIKHAA KATAKAM (U2223972L)
SUSHMITA RAMARATNAM (U2222958B)

TABLE OF CONTENTS

1

INTRODUCTION

PROBLEMS

ATTRITION OF EMPLOYEES WITH GREAT POTENTIAL

PROBLEMS THAT ARISE:

1. Lower Productivity and Losses
2. Increased Cost of Production as a result of hiring of new personnel and training costs

MOTIVATION & AIM

To determine the factors that contributed to the attrition of Watson healthcare's employees from their existing jobs

POTENTIAL APPLICATION

Through the analysis of the factors contributing to attrition of employees, companies can be more selective while hiring employees based on these factors.

Moreover, this also gives companies scope for improvement wherever applicable.

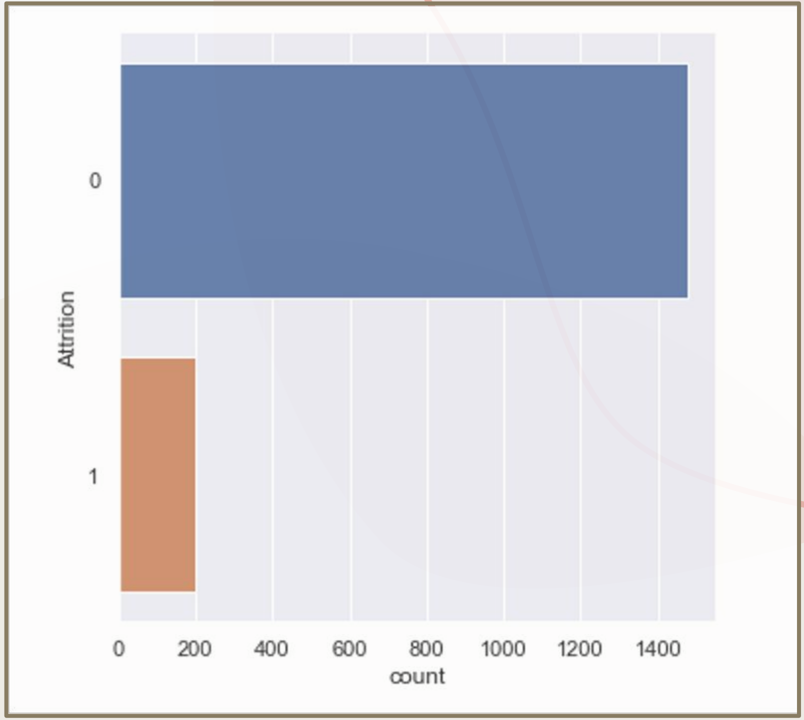
E.g. if lack of employee welfare was a reason for attrition, it could act as an indicator that it is about time for the company to implement the change.

DATA PREPROCESSING

STATISTICS

Total Number of Employees = 1676

11.9% of the employees left their jobs while 88.1% remained.



STEPS

Cleaning of Dataset → **Breaking down of Variables** → **Plotting of Relations**

The **removal** of insignificant columns from the dataset

Are they numeric/ categorical?

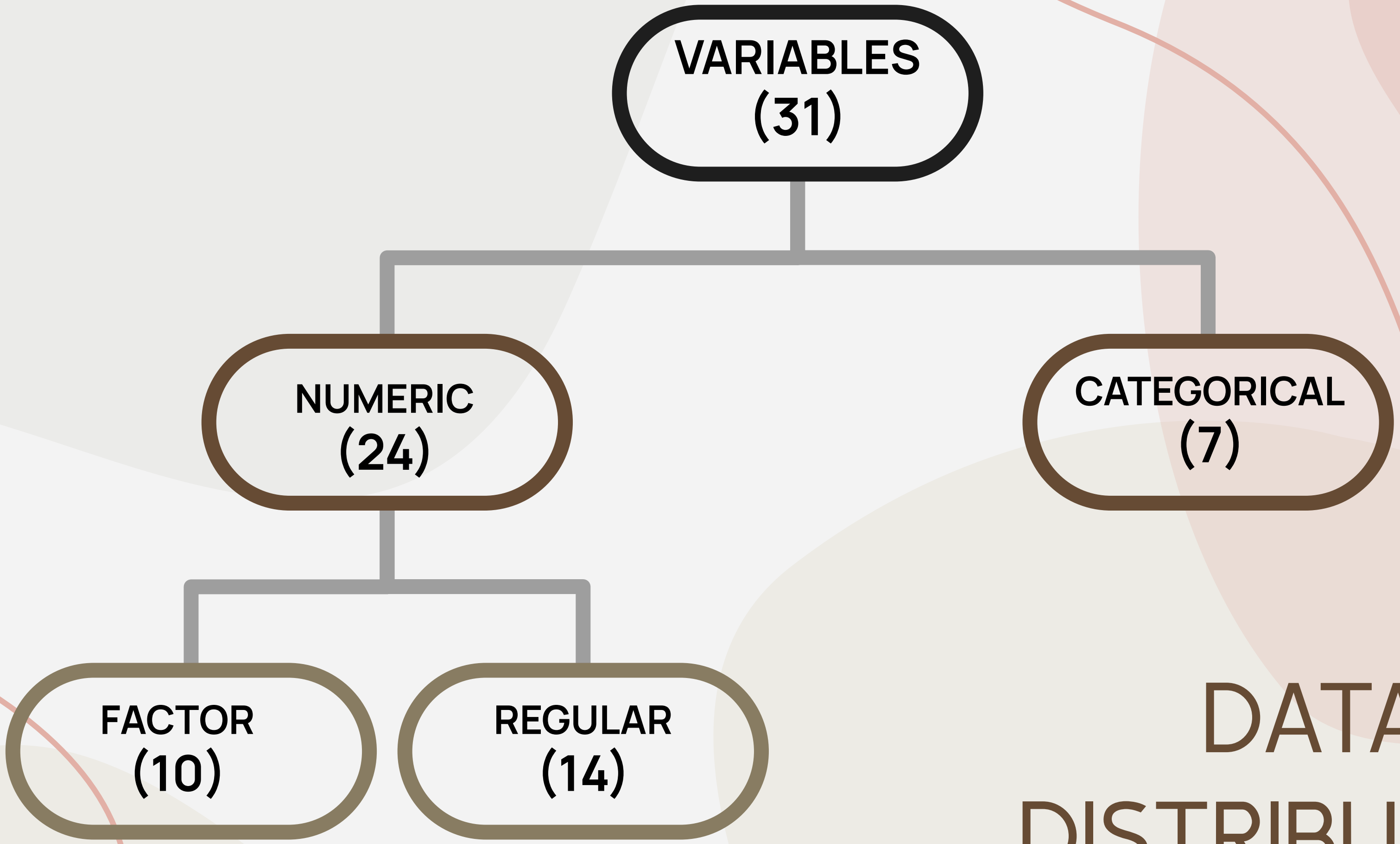
Which is the most appropriate model for the type of data?

CLEANING OF DATA

The columns below were deemed unnecessary/ redundant and hence removed:

1. **EmployeeCount** - all employees had the count 1
2. **Over18** - since it is standard practice to hire legal adults, all employees were above 18
3. **StandardHours** - all employees worked the standard hours
4. **EmployeeID** - the employee ID does not make any difference to their attrition since it is merely a means of identification

DATA DISTRIBUTION



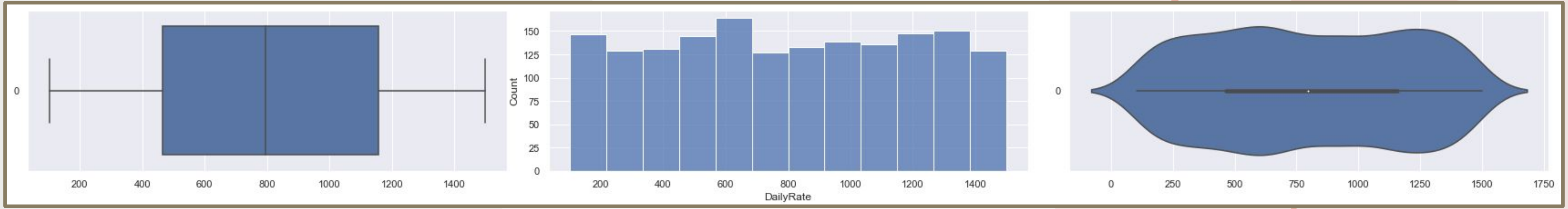
NUMERIC VARIABLES

For **Numeric Variables**, 3 types of graphs were implemented:

1. BoxPlot
2. Histogram Plot
3. Violin Plot

This is because of the following advantages:

1. **BoxPlot**: compact representation of distribution and show median,quartiles,outliers; easy to identify outliers
2. **Histogram Plot**: detailed view of distribution, useful to identify patterns
3. **Violin Plot**: summary statistics, shape of the distribution, show differences in the density of the distribution between groups

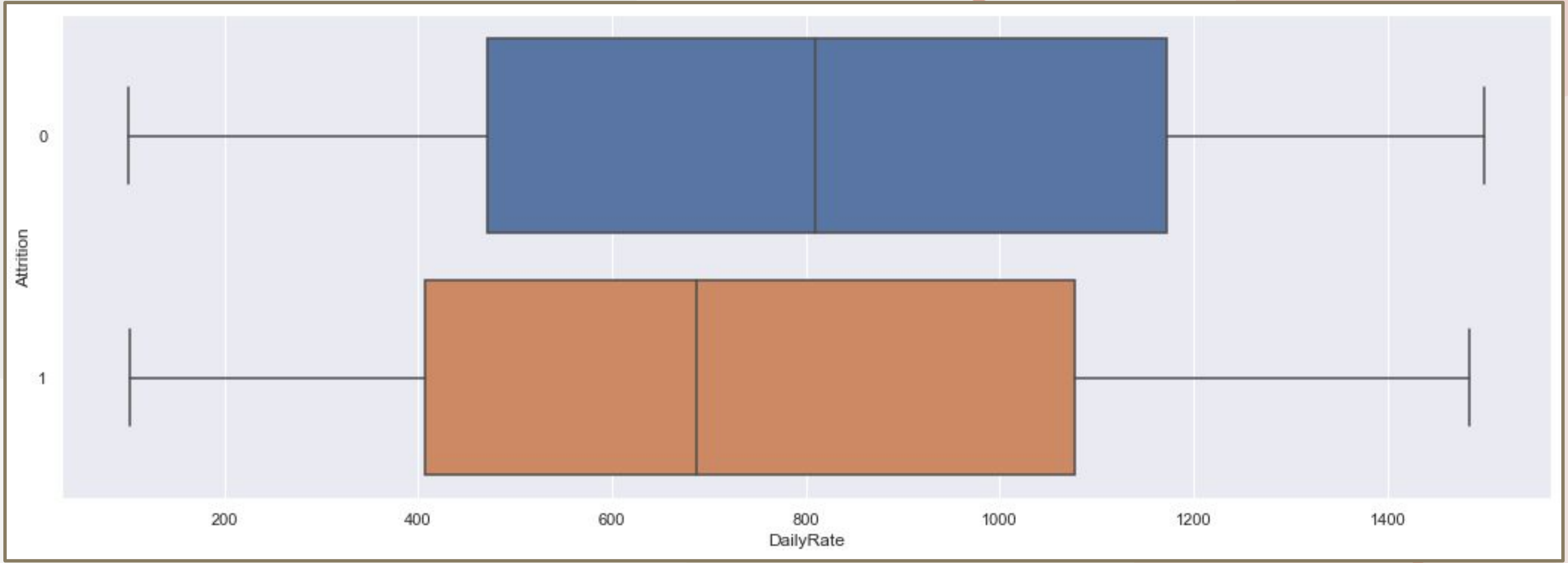


BoxPlot

Histogram Plot

Violin Plot

The above image depicts the BoxPlot, Histogram Plot and Violin Plot for the numeric variable ***DailyRate***



The above image depicts a BoxPlot between ***Attrition*** and ***DailyRate*** to provide a comparison.

CATEGORICAL VARIABLES

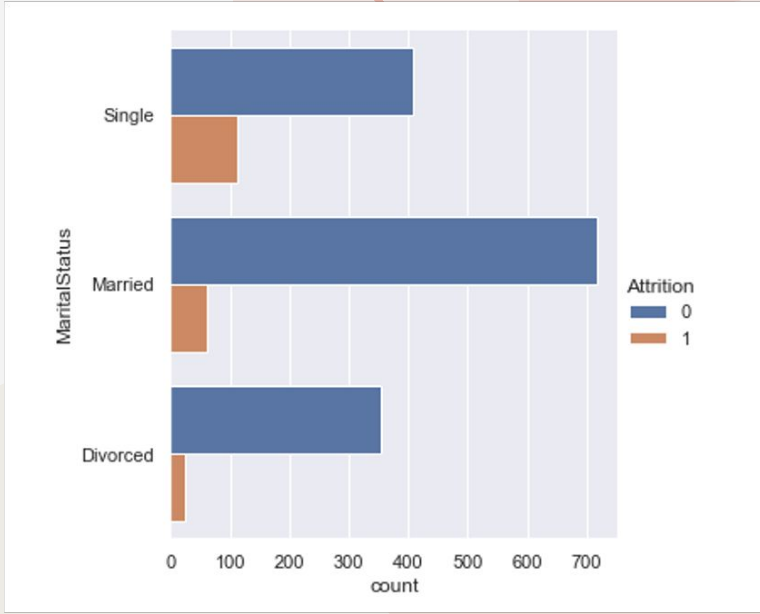
For **Categorical Variables**, the graph type implemented was **Bar Plot**.

This was done using GroupBy.

This is because of the following advantages:

1. **Easy interpretation:** height of each bar represents the frequency
2. **Category comparison:** useful for identifying patterns

This figure depicts the Bar Plot for the categorical variable ***MaritalStatus*** and ***Attrition*** to provide a comparison.

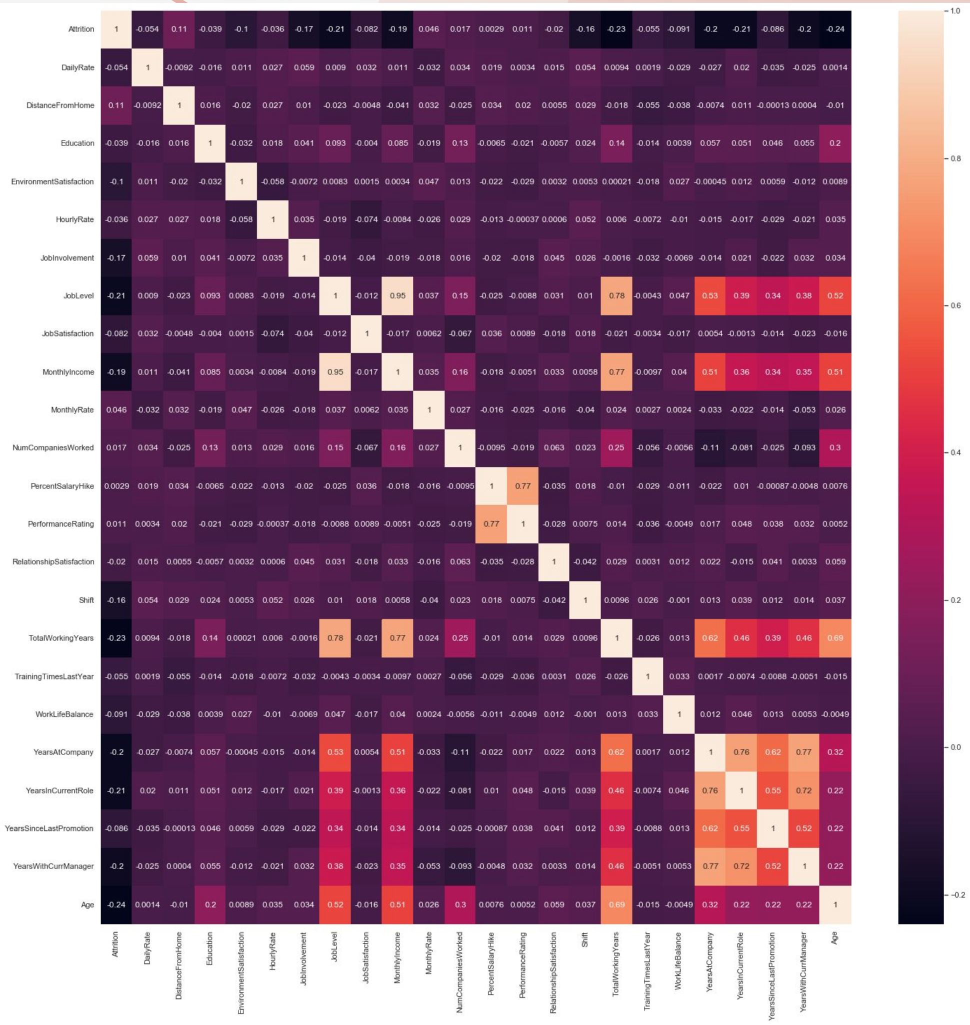


CORRELATION MATRIX

➔ Attrition has been re-classified as a numeric variable and is compared to other variables through a correlation matrix

➔ Strongest correlations:

- JobLevel vs TotalWorkingYears = **0.78**
- TotalWorkingYears vs MonthlyIncome = **0.77**
- YearsWithCurrManager vs YearsAtCompany = **0.77**
- YearsInCurrentRole vs YearsAtCompany = **0.76**



3

METHODOLOGY

METHODOLOGY

Machine Learning Model is a program that can make certain decision or find patterns from a previously unseen dataset.

The Machine Learning Model that we have chosen for our study is **Random Forest**

RANDOM FOREST

- Solves classification and regression problems
- Consists of many decision trees
- Establishes outcome based on the predictions of the decision trees

NOTE: **Decision Trees** are supervised learning models that predict the value of the target variable by learning simple decision rules inferred from data features

Why did we choose this model?

1. Provides an estimate of important variables
2. Accuracy and efficiency are high even in the case of large data sets
3. Can be saved and reused
4. Doesn’t overfit with more features unlike other models

How did we implement this model?

```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
```

- ★ Used to implement cross-validation and split data into train and test data sets
- ★ Used to calculate accuracy score for the model
- ★ Used to create random forest model

```
response = pd.read_csv("watson_healthcare_modified.csv",usecols = ['Attrition'])
predics = pd.DataFrame(healthdata[["MonthlyIncome", "Age", "YearsInCurrentRole",
                                   "TotalWorkingYears", "JobLevel","JobSatisfaction","EnvironmentSatisfaction",
                                   "JobInvolvement","YearsWithCurrManager","YearsAtCompany"]])
```

- ★ The response variable ('Attrition') is read and stored in a Pandas dataframe called "response".
- ★ A subset of predictor variables (e.g. MonthlyIncome, Age, etc.) are read and stored in a Pandas dataframe called "predics".

How did we implement this model?

```
train_predics,test_predics,
train_response,test_response = train_test_split(predics,response,test_size = 0.25, random_state = 42)
```

★ The predictor and response variables are split into training and testing datasets, with a test size of 0.25 (25% of the data) and a random state of 42

★ Returns four variables, which are assigned to “train_predics”, “test_predics”,” train_response”, and “test_response”

```
# import random forest and fit the data
rf = RandomForestClassifier(random_state=42,n_jobs=-1)
rf.fit(train_predics,train_response.values.ravel())
rf.fit(train_predics,train_response.values.ravel())

# use the current model and the test set of predictors to predict the response
response_pred = rf.predict(test_predics)
print("Accuracy: ",metrics.accuracy_score(test_response,response_pred))
```

★ Random Forest model is trained on the training data

★ Trained model is used to make predictions on test data

★ Accuracy of model’s predictions is compared to actual values

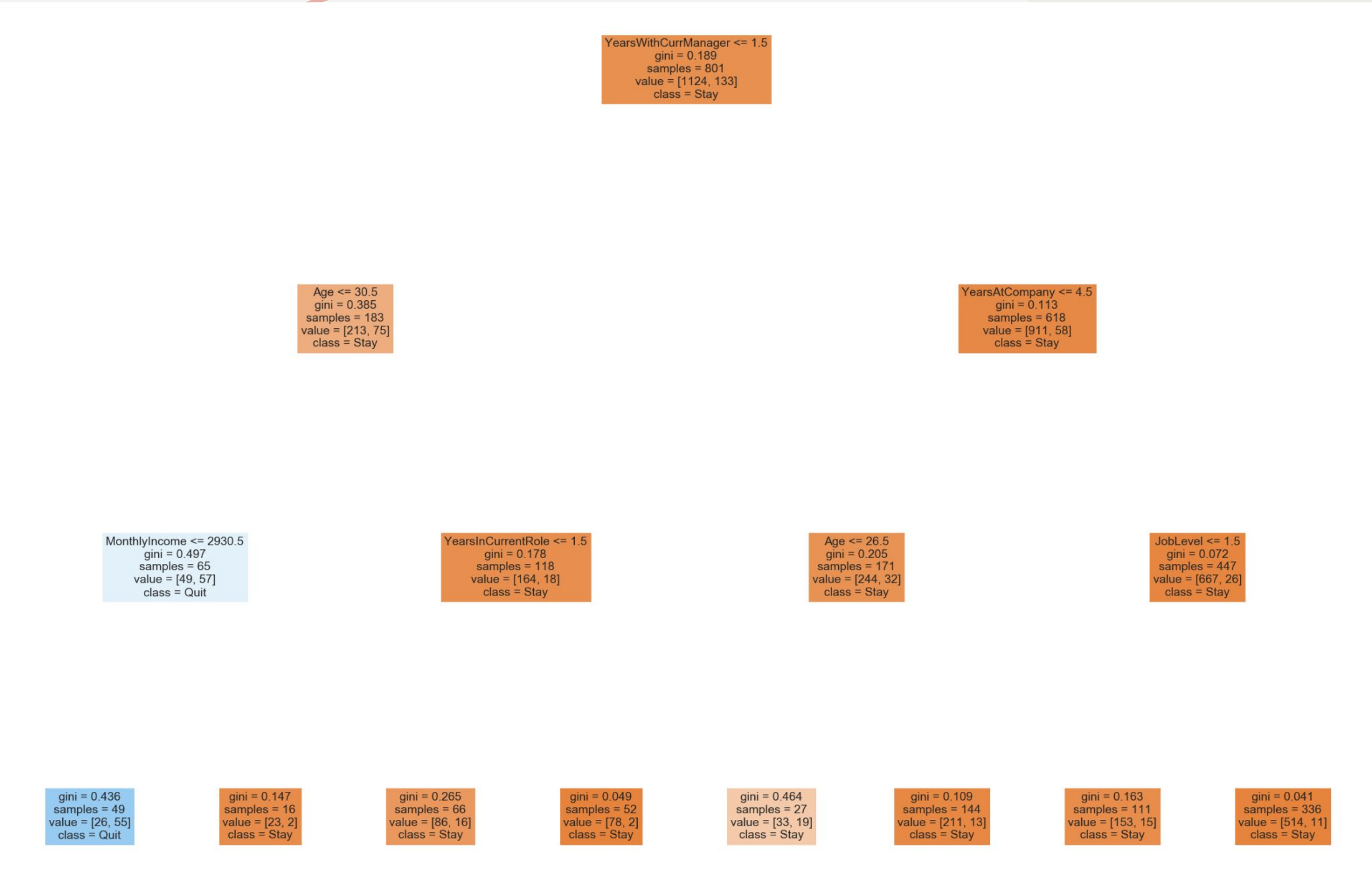
Generating a **Multivariate Decision Tree** from the Random Forest

```
# plot one of the trees from random forest
from sklearn.tree import plot_tree

randomized_best = random_search.best_estimator_

plt.figure(figsize=(40,30))

# tree created by estimators_[2], each tree is independent of each other
plot_tree(randomized_best.estimators_[2],feature_names = predics.columns,class_names=['Stay','Quit'],filled=True);
```



Accuracy = 90.13%

Decision Tree generated from our Random Forest

4

EXPERIMENTS

EXPERIMENTS

Metrics are used to measure and monitor the performance or quality of a model during training and testing.

We will be comparing our model with a linear regression model (**Logistic Regression**) and **Neural Network**.

LOGISTIC REGRESSION

What is Logistic Regression?

- statistical model that is used for **classification** and **predictive** analysis
- estimates the **probability** of an event occuring

Why use Logistic Regression?

1. simple machine learning algorithm and very efficient
2. outputs well-calibrated probabilities
3. gives inference about importance of every feature
4. updated easily to reflect new data

How did we implement this model?

```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
import warnings
warnings.simplefilter("ignore")
lr = LogisticRegression()

resp = pd.read_csv("watson_healthcare_modified.csv",usecols = ['Attrition'])

# random split the dataset into test and train
train_pred, test_pred, train_resp, test_resp = train_test_split(cate_pred,resp,test_size = 0.25, random_state = 42)

# fit the logistic regression model with train dataset
lr.fit(train_pred,train_resp)

train_accuracy = lr.score(train_pred,train_resp)
print('Accuracy on the train set: {:.2f}'.format(train_accuracy))

resp_pred = lr.predict(test_pred)
test_accuracy = accuracy_score(test_resp,resp_pred)
print('Accuracy on the test set: {:.2f}'.format(test_accuracy))
```

```
Accuracy on the train set: 0.90
Accuracy on the test set: 0.86
```

Logistic Regression - Inference

- determine the **level of influence** of categorical variables on attrition
- **convert** the variables into numeric indicator variables with get_dummies
- **Accuracy**: 0.90 (Train), 0.86 (Test)
  - The model only classifies 61% of employees that quit, correctly

	OverTime	Gender	MaritalStatus	Department	EducationField
0	Yes	Female	Single	Cardiology	Life Sciences
1	No	Male	Married	Maternity	Life Sciences
2	Yes	Male	Single	Maternity	Other
3	Yes	Female	Married	Maternity	Life Sciences
4	No	Male	Married	Maternity	Medical
...	...	...	...	...	...
1671	Yes	Male	Single	Neurology	Technical Degree
1672	Yes	Female	Married	Cardiology	Marketing
1673	No	Female	Single	Maternity	Life Sciences
1674	No	Female	Married	Neurology	Life Sciences
1675	No	Female	Single	Cardiology	Medical

NEURAL NETWORK

What is a Neural Network?

Type of machine learning model inspired by the structure and function of biological neurons in the human brain

Why use Neural Networks?

1. Can capture **non-linear relationships** between input features and output variables - can model complex patterns
2. Can handle **high-dimensional data**
3. **Robustness to noise and missing data** - can learn to ignore irrelevant or missing features and focus on the most important ones.
4. Can **adapt to changing data** and learn from new examples

How did we implement this model?

```
# start training the model

num_epochs = 3

train_loss = []
test_loss = []
train_accuracy = []
test_accuracy = []

for epoch in range(num_epochs):

    train_correct = 0
    train_total = 0

    for i, (items, classes) in enumerate(train_loader):

        items = Variable(items)
        classes = Variable(classes)

        # Put the model in training mode
        net.train()

        # Calculate the loss and gradients
        optimizer.zero_grad()
        outputs = net(items)
        loss = criterion(outputs, classes.to(torch.int64))
        loss.backward()
        optimizer.step()

        # Record the correct predictions for training data
        train_total += classes.size(0)
        _, predicted = torch.max(outputs.data, 1)
        train_correct += (predicted == classes.data).sum()

        print ('Epoch %d/%d, Iteration %d/%d, Loss: %.4f'
               %(epoch+1, num_epochs, i+1, (len(nn_train)//100)+1, loss.data.item()))
```

```
    # Model in evaluation mode
    net.eval()

    # Record the loss and train accuracy
    train_loss.append(loss.data.item())
    train_accuracy.append((100 * train_correct / train_total))

    # Record the correct predictions for test data
    test_items = torch.FloatTensor(nn_test.values[:, 0:3])
    test_classes = torch.LongTensor(nn_test.values[:, 3])

    # Record the test accuracy
    outputs = net(Variable(test_items))
    loss = criterion(outputs, Variable(test_classes))
    test_loss.append(loss.data.item())
    _, predicted = torch.max(outputs.data, 1)
    total = test_classes.size(0)
    correct = (predicted == test_classes).sum()
    test_accuracy.append((100 * correct / total))
```

```
Epoch 1: train accuracy = 76.42, test accuracy = 86.01
Epoch 2: train accuracy = 88.66, test accuracy = 86.01
Epoch 3: train accuracy = 88.66, test accuracy = 86.01
```

Avg. Train Accuracy = **84.58%**

Avg. Test Accuracy = **86.01%**

CONCLUSION

1. Random Forest
2. Logistic Regression
3. Neural Network

Data Driven Insights

Reasons for Leaving

❑ MonthlyIncome	: Lower income employees tend to leave
❑ Age	: Younger employees tend to leave
❑ DistanceFromHome	: Employees living further from Home tend to leave
❑ TotalWorkingYears	: Employees who have worked lesser tend to leave
❑ YearsAtCompany	: Employees who have worked lesser tend to leave
❑ YearsInCurrentRole	: Employees who have worked longer in a role tend to leave
❑ YearsWithCurrManager	: Employees who have worked lesser tend to leave
❑ EnvironmentSatisfaction	: Employees less satisfied with environment tend to leave
❑ JobSatisfaction	: Employees less satisfied with job tend to leave

DATASET: https://www.kaggle.com/datasets/jpmiller/employee-attrition-for-healthcare

REFERENCES

https://www.databricks.com/glossary/machine-learning-models#:~:text=Some%20popular%20examples%20of%20machine,%2C%20random%20forest%2C%20and%20XGBoost.

https://scikit-learn.org/stable/modules/tree.html

https://neptune.ai/blog/performance-metrics-in-machine-learning-complete-guide#:~:text=Performance%20metrics%20are%20a%20part,a%20metric%20to%20judge%20performance.

https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

https://www.mygreatlearning.com/blog/random-forest-algorithm/

https://www.ibm.com/topics/logistic-regression

Thank You!

slidesgo