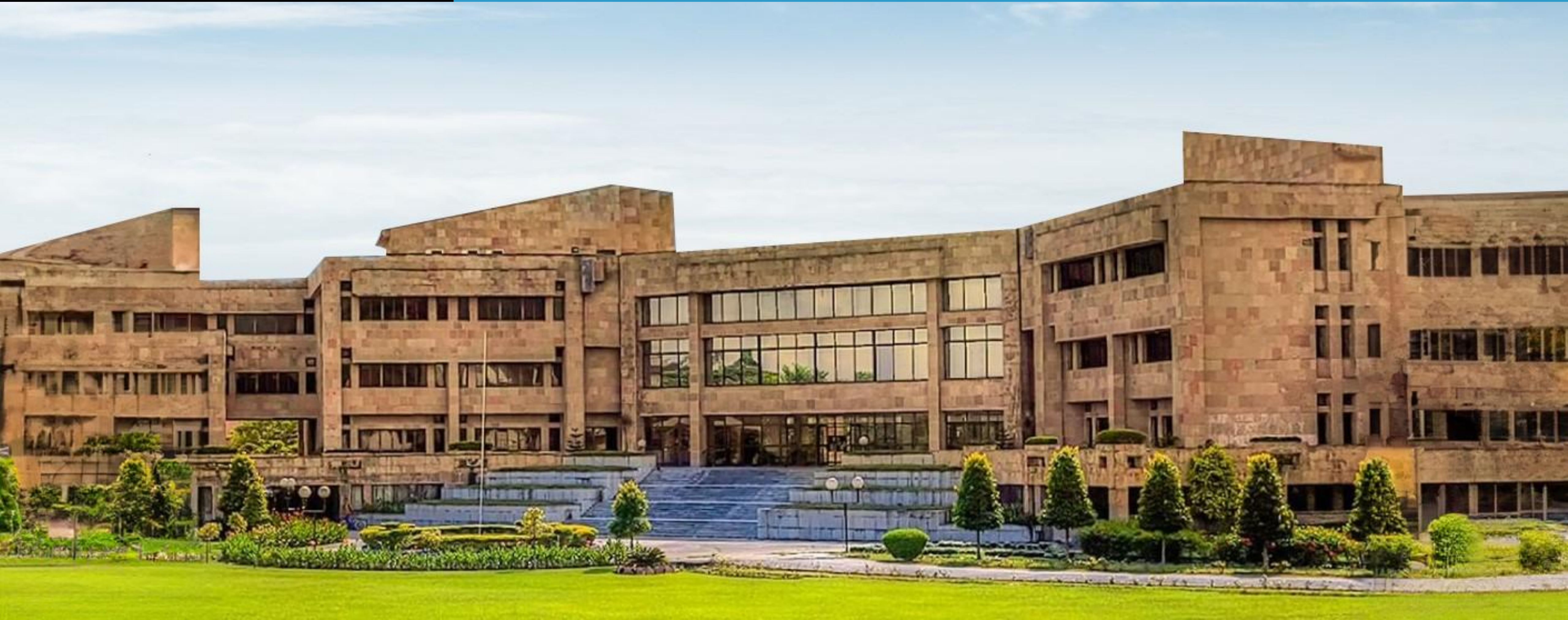
E & C
IIT Guwahati

Advanced Certification Programme in Data Science Business Analytics



Week 20
Predictive Modelling

Topics Covered

- Implementing Linear Regression
- Implementing Logistic Regression
- Q & A

Async Recap

1. Understand Data Science Foundations
Master core mathematical and statistical concepts like linear algebra, probability, and descriptive statistics to support data discovery and prediction.

2. Apply Linear Algebra to Data
Use vectors, matrices, and transformations to handle complex data structures and optimise machine learning models.

3. Use Key Operations in Data Handling
Perform essential vector and matrix operations such as addition, multiplication, and transposition for efficient data analysis.

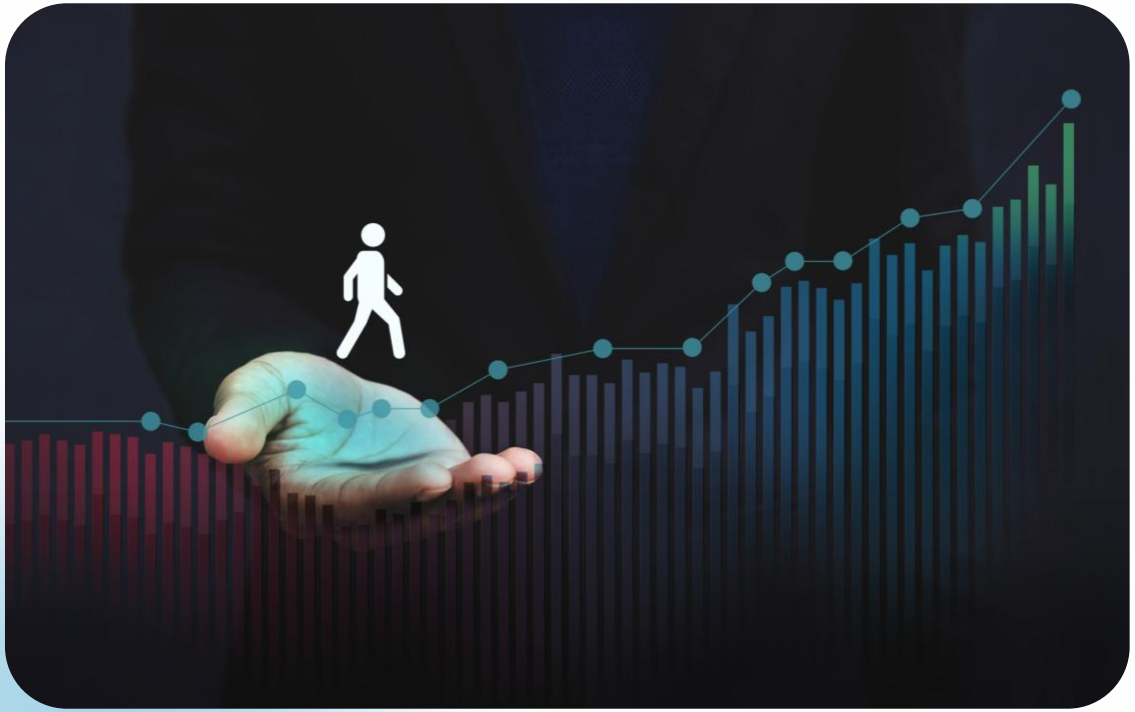
4. Summarise and Visualise with Statistics
Describe data using statistical summaries and visual tools like bar, pie, and box plots to identify trends and patterns.

5. Assess Risk Using Probability
Quantify uncertainty with probability to support decision-making and evaluate event likelihoods.

Implementing Linear Regression

Introduction to Linear Regression with Statsmodels

Modelling Relationships and Analysing Predictors



- Model the relationship between dependent and independent variables
- Provide detailed statistical summaries
- Support hypothesis testing and confidence intervals
- Prefer for in-depth regression over Scikit-Learn
- Analyse effect of multiple predictors on house prices

Dataset Overview

Features Used to Predict House Price

Feature	Description
Area	Square footage of the house
Bedrooms	Number of bedrooms
Bathrooms	Number of bathrooms
Material	Type of construction material (Concrete/Masonry)
Locality	Location of the house
Price	House price (dependent variable)

- Build a linear regression model to predict price based on features

Importing Libraries and Loading Data

Preparing Data for Linear Regression

Code implementation

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from sklearn.preprocessing import
OneHotEncoder

# Load dataset
data = pd.read_csv('house.csv')

# One-hot encode categorical variables
data = pd.get_dummies(data,
                columns=['Material', 'Locality'],
                drop_first=True)
```

Selecting features and adding a constant term

```
# Selecting features and target variable

X = data[['Area', 'Bedrooms', 'Bathrooms',
       'Material_Masonry', 'Locality_Summit View']]
y = data['Price']

# Adding constant term for intercept
X = sm.add_constant(X)
```

Building and Fitting the Model

Creating and Training an OLS Regression Model

Build and fit model

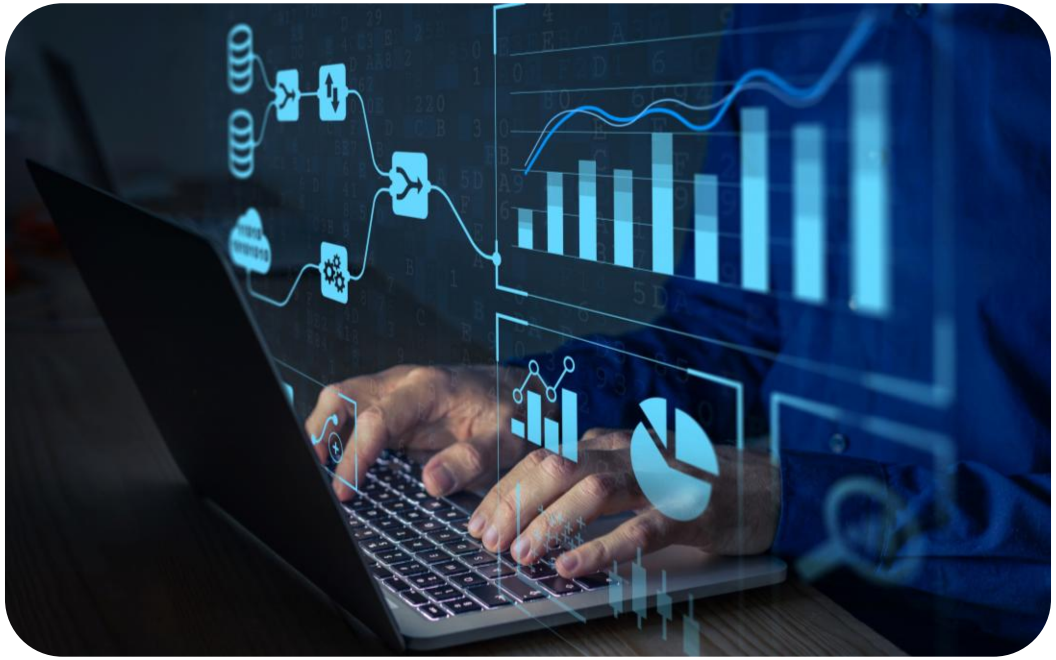
- Create OLS model and fit it to the data

```
# Building the model
model = sm.OLS(y, X)

# Fitting the model
results = model.fit()
```

Model Summary and Interpretation

Understanding Key Outputs from the Regression Model



- **R-squared**: Measures goodness of fit
- **F-statistic and p-value**: Determines overall model significance
- **Coefficients and standard errors**: Indicates impact and reliability of predictors
- **P-values**: Test significance of individual predictors

```
# Model summary
print(results.summary())
```

Evaluating Model Performance

Assessing Predictions and Error Metrics

Make predictions

Generate predicted values using the fitted model

```
# Making predictions

y_pred = results.predict(X)
```

Calculate residuals

Compute error between actual and predicted

```
# Calculating residuals

residuals = y - y_pred
```

Compute error metrics

Calculate MSE and RMSE to evaluate model performance

```
from sklearn.metrics import mean_squared_error

mse = mean_squared_error(y, y_pred)

rmse = np.sqrt(mse)
print(f'MSE: {mse}, RMSE: {rmse}')
```

Checking Regression Assumptions

Ensuring Validity of Model Results

Linearity:

- Ensure relationship between variables is linear
- Check with scatterplot of residuals vs predicted values

Independence:

- Ensure residuals are independent
- Use Durbin-Watson test

Homoscedasticity:

- Ensure constant variance in residuals
- Use Breusch-Pagan test

Normality:

- Ensure residuals follow normal distribution
- Use QQ plot and Shapiro-Wilk test

Handle assumption violations:
Use polynomial regression for non-linearity, remove correlated predictors, log-transform for heteroscedasticity, and apply robust regression for non-normal residuals.

Checking Regression Assumptions

```
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.stats.diagnostic import het_breuschpagan
from scipy.stats import shapiro

# Linearity check
sns.residplot(x=y_pred, y=residuals, lowess=True)
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.title('Residual Plot')
plt.show()

# Normality check (QQ plot)
sm.qqplot(residuals, line='45')
plt.title('QQ Plot of Residuals')
plt.show()

# Shapiro-Wilk test
shapiro_test = shapiro(residuals)
print(f'Shapiro-Wilk Test p-value: {shapiro_test.pvalue}')

# Homoscedasticity test
_, pval, _, _ = het_breuschpagan(residuals, X)
print(f'Breusch-Pagan Test p-value: {pval}')
```

Implementing Logistic Regression

Logistic Regression with Statsmodels

Modelling Binary Outcomes with Statistical Insight

- Model binary outcomes based on predictors
- Provide detailed statistical summaries
- Support hypothesis testing and confidence intervals
- Interpret logistic regression results effectively
- Predict Titanic survival using passenger features

Dataset Overview

Features Used to Predict House Price

Variable	Description
Survived	Binary target variable (1 = Survived, 0 = Did Not Survive)
Pclass	Passenger class (1st, 2nd, 3rd)
Sex	Gender of passenger
Age	Age of passenger
SibSp	Number of siblings/spouses aboard
Parch	Number of parents/children aboard
Fare	Ticket fare
Embarked	Port of embarkation (C, Q, S)

- Build a linear regression model to classify whether a passenger survived or not

Importing Libraries and Loading Data

Preparing Titanic Dataset for Logistic Regression

Import Necessary Libraries:
- Use pandas, NumPy and Statsmodels for data handling and modelling

Import Preprocessing Tools:
- Use OneHotEncoder for categorical encoding

Load the Dataset:
- Read Titanic data from CSV into pandas DataFrame

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from sklearn.preprocessing import
OneHotEncoder

# Load dataset
data = pd.read_csv('titanic.csv')
```

Handling Missing Values and Data Preprocessing

Preparing Clean and Structured Input for the Model

 # Fill missing values

```
data['Age'].fillna(data['Age'].median(), inplace=True)
```

 # Drop irrelevant columns

```
data.drop(columns=['PassengerId', 'Name', 'Ticket', 'Cabin'], inplace=True)
```

 # One-hot encode categorical variables

```
data = pd.get_dummies(data, columns=['Sex', 'Embarked'], drop_first=True)
```

Selecting Features and Building the Model

Preparing and Training a Logistic Regression Model

```
# Select features and target
X = data[['Pclass', 'Age', 'SibSp', 'Parch', 'Fare', 'Sex_male', 'Embarked_Q', 'Embarked_S']]
y = data['Survived']

# Add constant term
X = sm.add_constant(X)

# Build the model
model = sm.Logit(y, X)

# Fit the model
results = model.fit()
```

Model Summary and Interpretation

Evaluating Logistic Regression Results

- **Log-likelihood:** Measure model fit
- **Pseudo R-squared:** Use as an alternative to R-squared
- **Coefficients and odds ratios:** Interpret effect of each predictor
- **P-values**: Determine predictor significance
- **Interpretation Example**:
  Odds ratio > 1 increases survival probability
  Odds ratio < 1 decreases survival probability

```
# Code Implementation


print(results.summary())
odds_ratios = np.exp(results.params)
print(odds_ratios)
```

Model Evaluation and Assumption Checks

Assessing Predictions and Ensuring Model Validity

Make predictions

```
y_pred_prob = results.predict(X)

y_pred = [1 if prob > 0.5 else 0 for prob in y_pred_prob]
```

Calculate metrics

```
confusion_matrix(y, y_pred)

accuracy_score(y, y_pred)
```

Check assumptions

- Linearity of log-odds
- Independence of observations
- No perfect multicollinearity
- Large sample size

Handle violations

- Use polynomial terms for non-linearity
- Remove correlated predictors
- Adjust for class imbalance

Q & A

Thank you