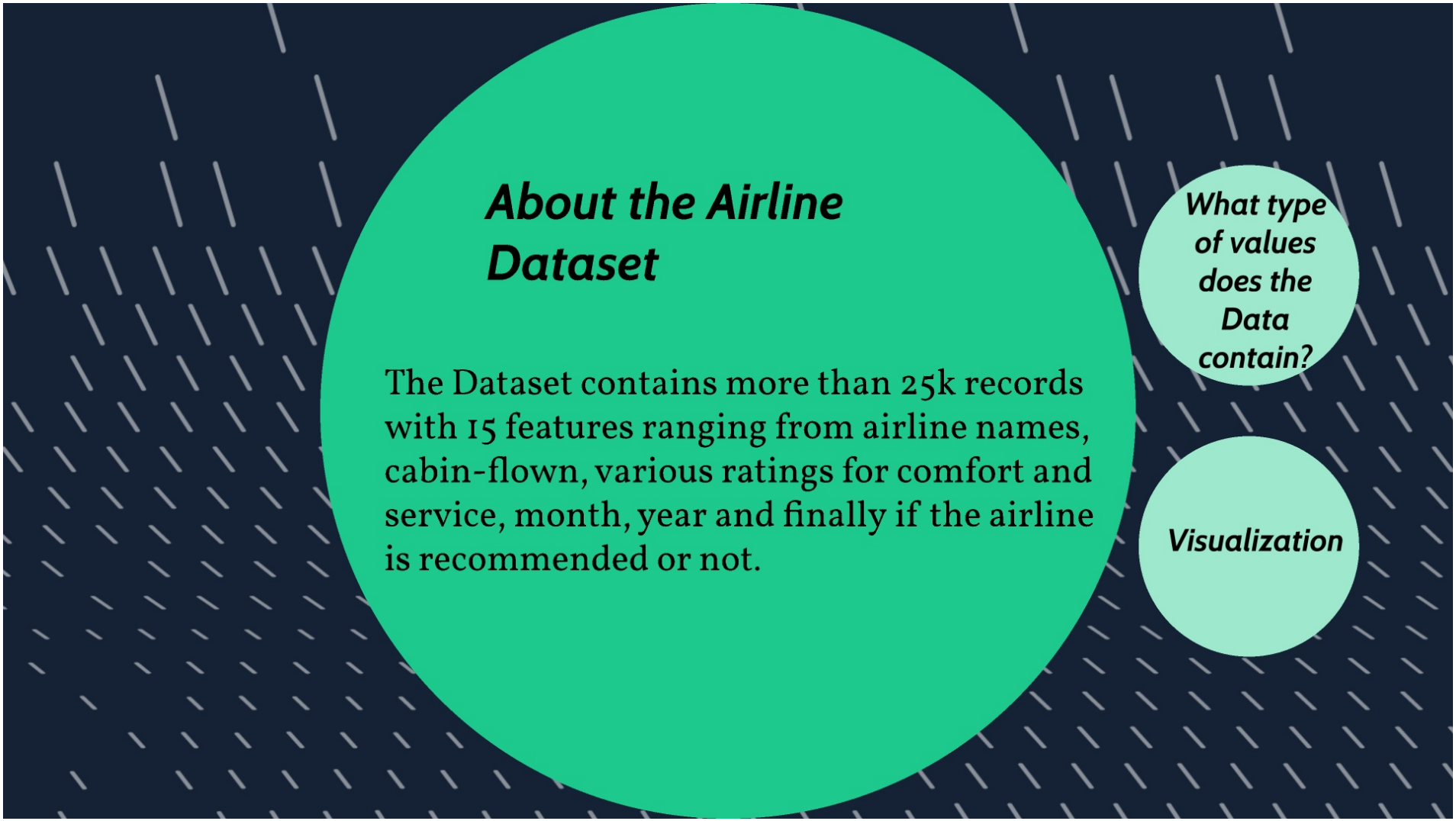
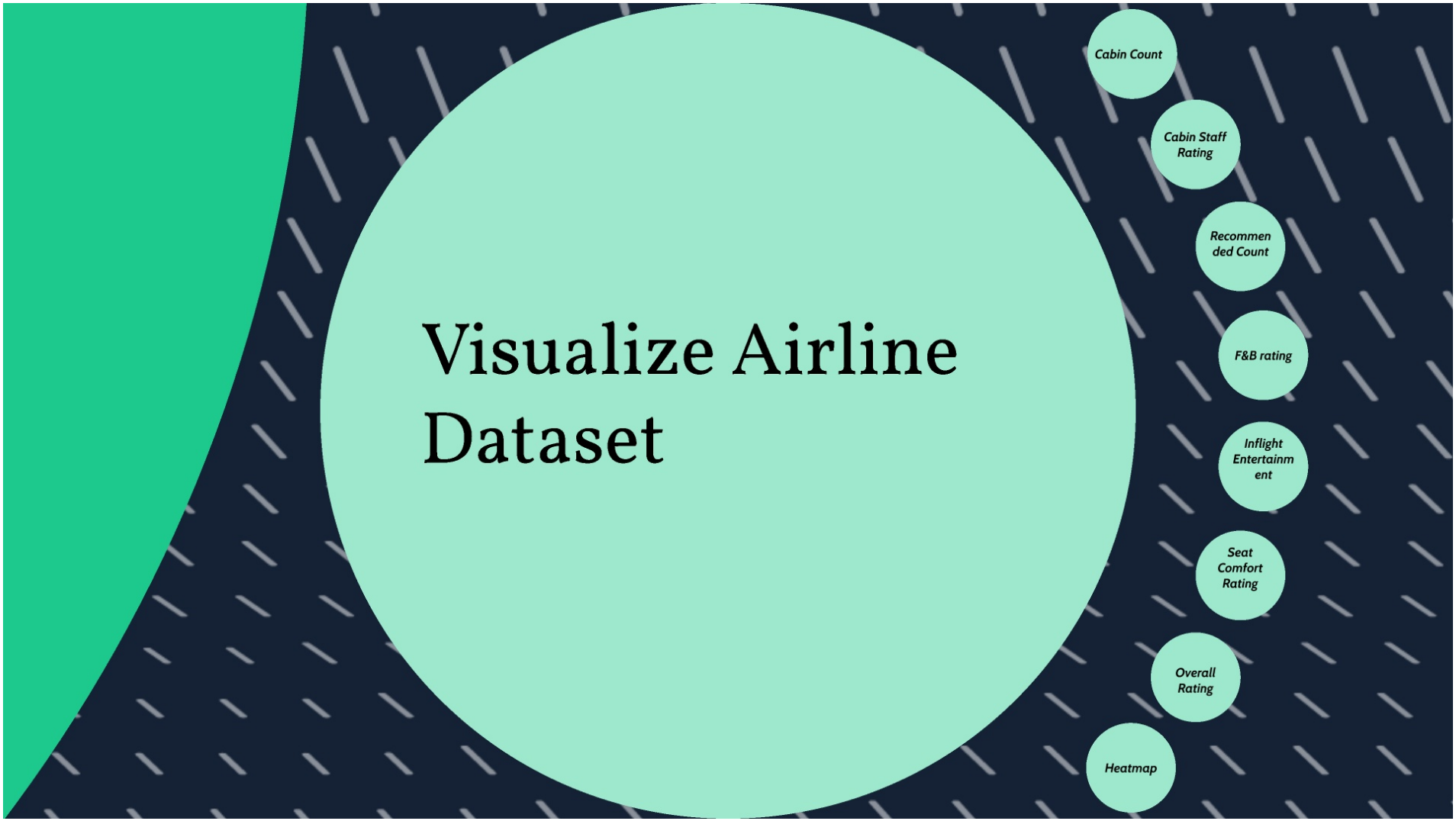

Introduction
Classification
Clustering
Thank You!
Data Mining on Airline Reviews Dataset


About the Airline Dataset
The Dataset contains more than 25k records with 15 features ranging from airline names, cabin-flown, various ratings for comfort and service, month, year and finally if the airline is recommended or not.
What type of values does the Data contain?
Visualization

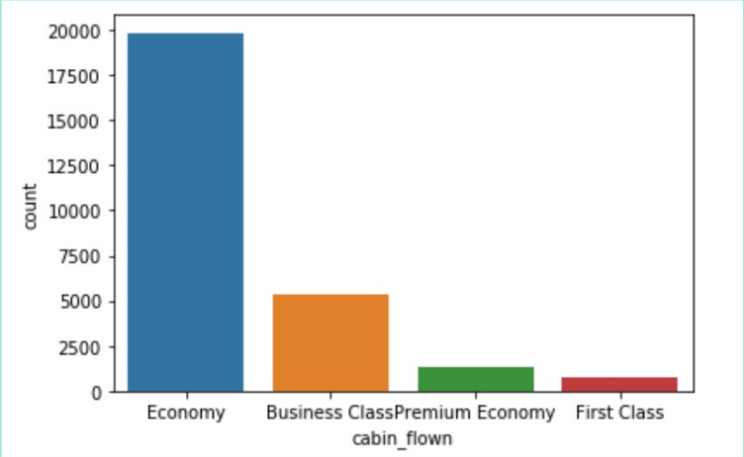
What type of values does the Data contain?

The airline review dataset contains both categorical values and numerical values such as:

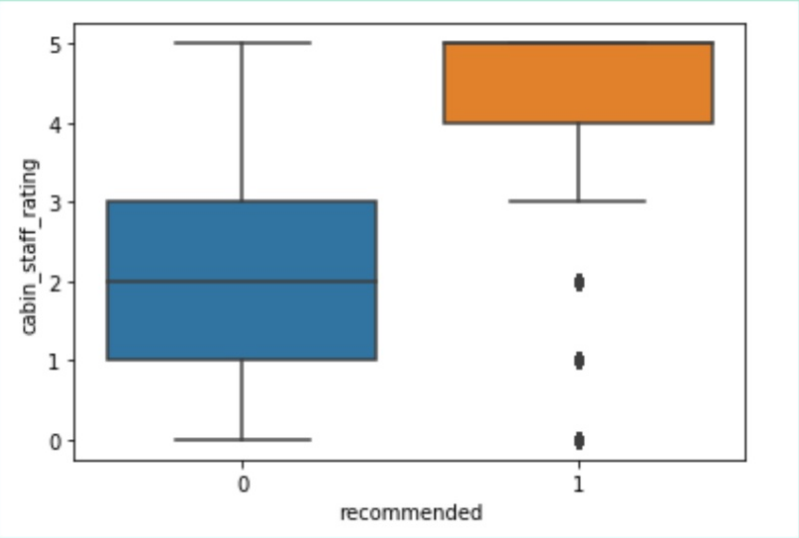
- Values for all the rating features ranges from 0-10.
- There are 292 unique categorical airline names.
- 4 categorical cabin class values.
- A "recommended" column which contains either a "0" or a "1". This is our target variable.


Visualize Airline Dataset
Cabin Count
Cabin Staff Rating
Recommended Count
F&B rating
Inflight Entertainment
Seat Comfort Rating
Overall Rating
Heatmap

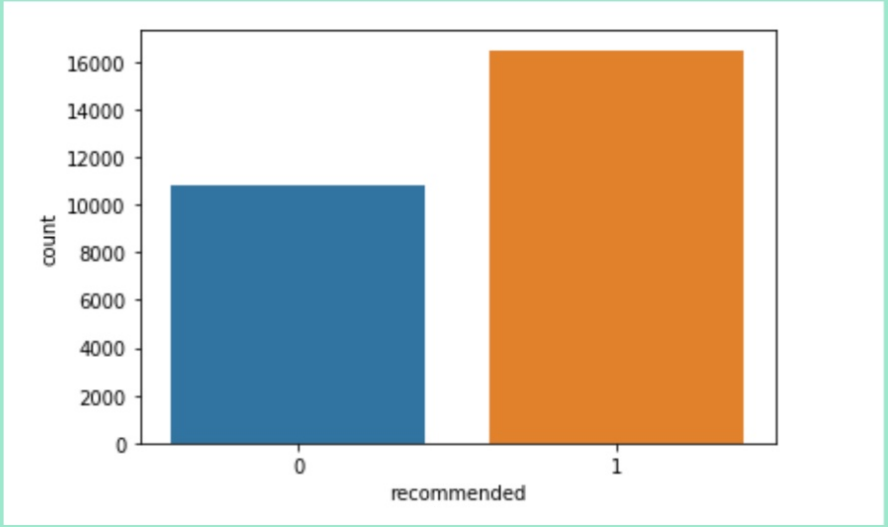
Cabin Count


20000
17500
15000
12500
10000
7500
5000
2500
0
count
Economy
Business Class
Premium Economy
First Class
cabin_flown

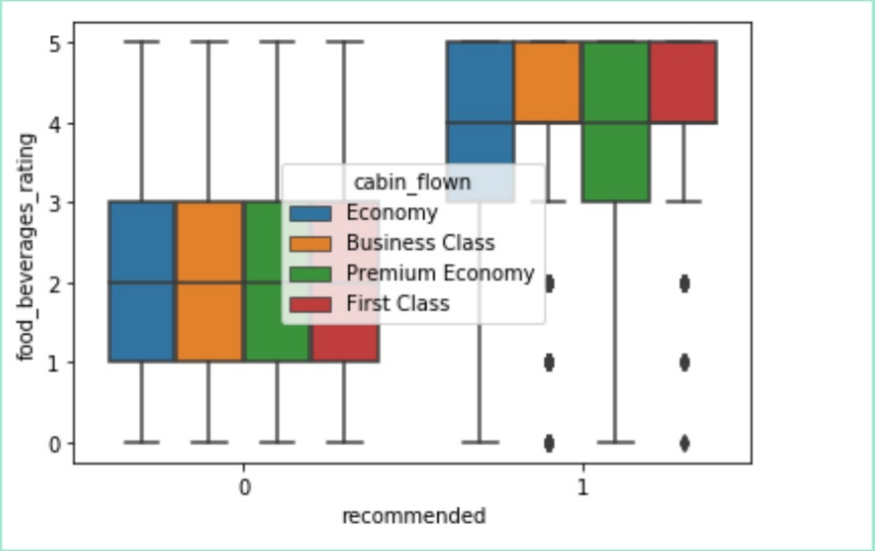
Cabin Staff Rating



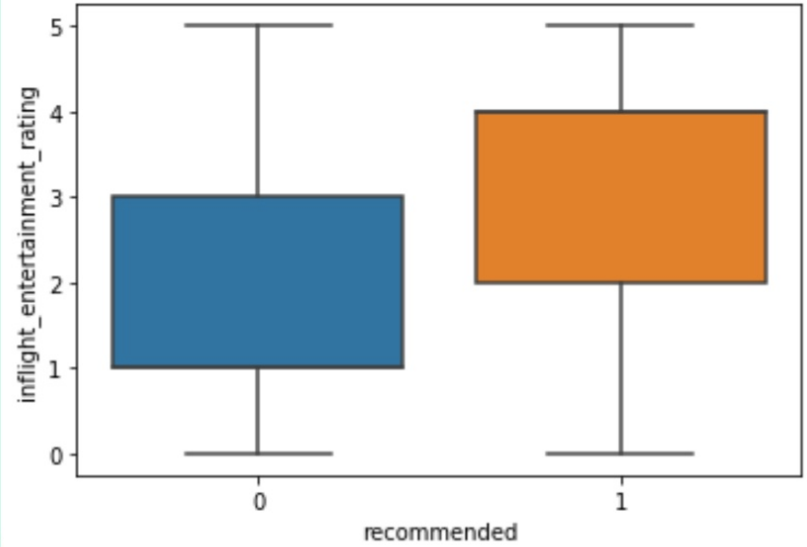
Recommended Count



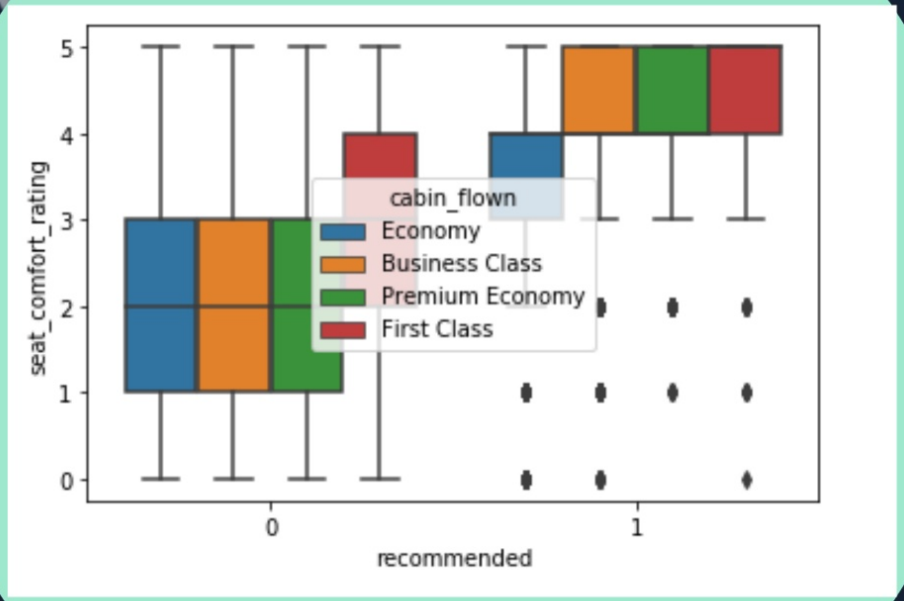
F&B Rating


food_beverages_rating
recommended
cabin_flown
Economy
Business Class
Premium Economy
First Class
0
1
2
3
4
5
0
1

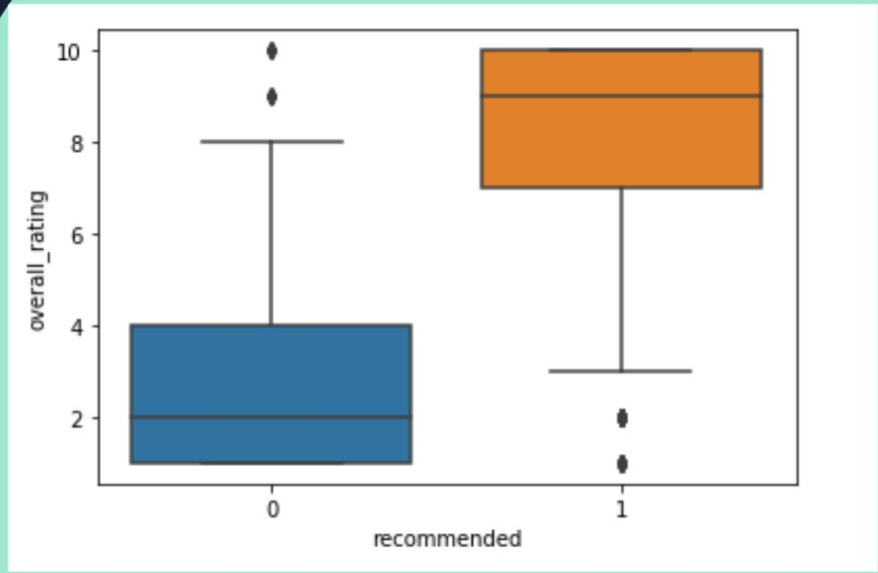
Inflight Entertainment Rating



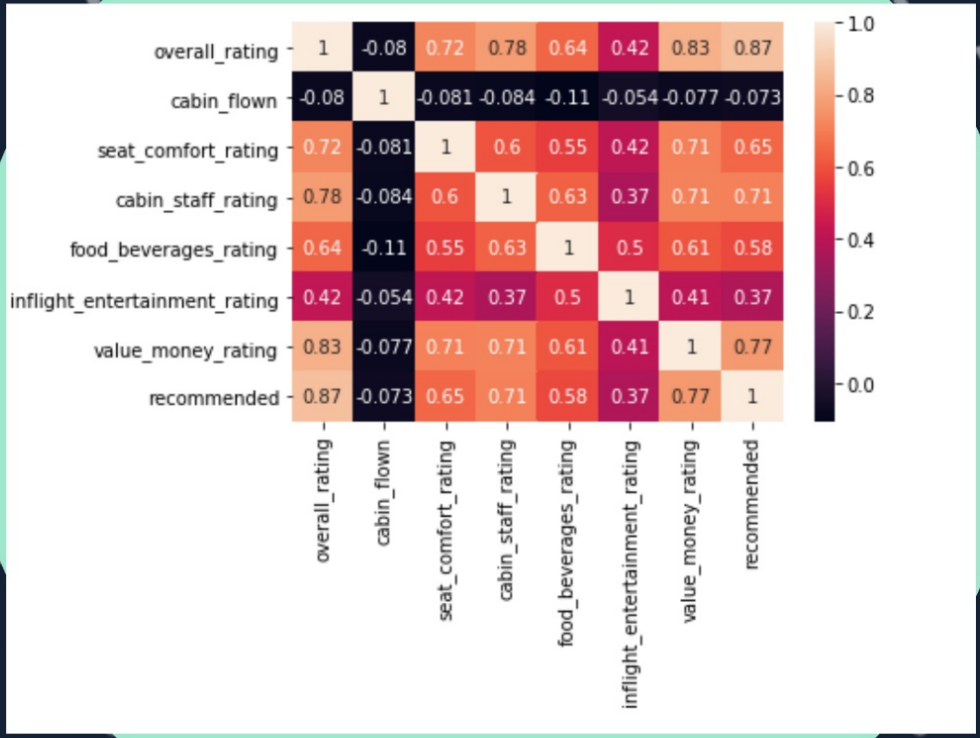
Seat Comfort Rating



Overall Rating



Heat Map


overall_rating
cabin_flown
seat_comfort_rating
cabin_staff_rating
food_beverages_rating
inflight_entertainment_rating
value_money_rating
recommended
1 -0.08 0.72 0.78 0.64 0.42 0.83 0.87
-0.08 1 -0.081 -0.084 -0.11 -0.054 -0.077 -0.073
0.72 -0.081 1 0.6 0.55 0.42 0.71 0.65
0.78 -0.084 0.6 1 0.63 0.37 0.71 0.71
0.64 -0.11 0.55 0.63 1 0.5 0.61 0.58
0.42 -0.054 0.42 0.37 0.5 1 0.41 0.37
0.83 -0.077 0.71 0.71 0.61 0.41 1 0.77
0.87 -0.073 0.65 0.71 0.58 0.37 0.77 1
1.0
0.8
0.6
0.4
0.2
0.0


Introduction
Classification
Clustering
Thank You!
Data Mining on Airline Reviews Dataset


Clustering
- Clustering is grouping set of objects based on characteristics & aggregating them according to their similarities.
- Clustering is a Machine Learning technique that involves the grouping of data points.
PCA
K-Means

K-Means

1. Select a number of classes/groups to use.
2. Randomly initialize their respective center points.
3. Each data point is classified by computing the distance between that point and each group center, and then classifying the point to be in the group whose center is closest to it.
4. Recompute the group center by taking the mean of all the vectors in the group.
5. Repeat these steps for a set number of iterations or until the group centers don't change much between iterations.

Advantages:
- K-Means is pretty fast.
- linear complexity O(n*n)

Disadvantages:
- Select number of groups/classes in advance.
- Results may not be repeatable and lack consistency as we randomly select centroids.

PCA

- Principal Component Analysis (PCA) is an unsupervised machine learning technique.

- PCA is a popular technique for deriving a set of low dimensional features from large set of variables.

PCA & K-Means Clustering on Airline Dataset

PCA & K-Means Clustering on Airline Dataset

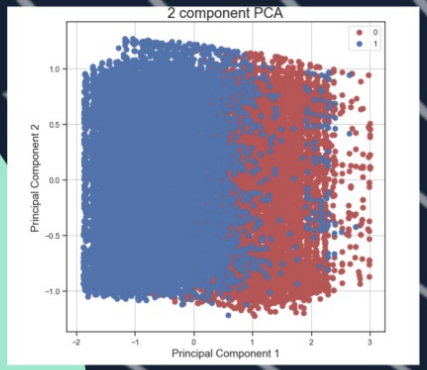
Steps:

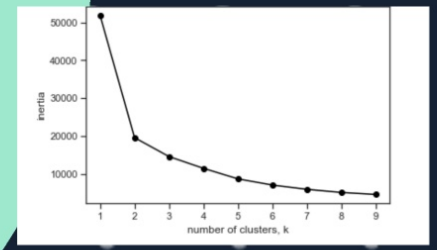
1. Data Preprocessing-
   i. Label Encoding of Categorical features
   ii. Min-Max Scaler
2. from sklearn.decomposition import PCA
3. Number of Components = 2

```
1 from sklearn.decomposition import PCA
2 pca = PCA(n_components=2)
3 principalComponents = pca.fit_transform(x_scaled)
4 principalDf = pd.DataFrame(data = principalComponents, columns = ['principal component 1'
5 principalDf
```

	principal component 1	principal component 2
0	-0.181938	1.187181
1	-1.019887	1.217195
2	-0.880514	1.247218
3	-0.304972	1.163973
4	1.014821	0.962267
...	...	...
27279	0.561985	-0.756752
27280	1.453114	-0.868342
27281	2.054904	-0.892463
27282	0.353570	-0.841416
27283	1.587505	-0.872256

27284 rows × 2 columns






Introduction
Classification
Clustering
Thank You!
Data Mining on Airline Reviews Dataset


Naive Bayes
Classification
- Classic Data Mining Technique
- Classify each item in a set of data into one of a predefined set of classes or groups.
Random Forests
Decision Trees
SVM

How does Decision Trees work?

Advantages & Disadvantages

Decision Tree on Airline Dataset

Decision Trees

- Supervised Machine Learning

- Data is continuously split based on certain parameters

- Decision Tree consists of :

> **Nodes** : Attributes value & it's condition.

> **Edges/ Branch** : Correspond to the outcome of a test and connect to the next node or leaf.

> **Leaf nodes** : Terminal nodes that predict the outcome or the Target variables & it's value.

How does Decision Trees work?

- Start the **tree root** and **split** the data on the **feature** that results in the largest **information gain (IG)**
- **Repeat** this splitting procedure at each child node until the **leaves are pure** i.e. samples at each leaf node all belong to the same class.
- Generally, set limit to tree depth to avoid over fitting.

Advantages & Disadvantages

Advantages :

- *Easy to interpret* for small-sized trees.
- Excludes unimportant features.
- *Accuracy* comparable to other classification techniques for many simple data sets.
- *Extremely fast* at classifying unknown records.
- Inexpensive to construct.

Disadvantages :

- *Easy to over fit*.
- Adding training data can result in large changes to decision logic.
- *Large trees can be difficult to interpret* and the decisions they make may seem counter intuitive.

Decision Tree on Airline Dataset

Steps:

1. Data Pre-processing -
   i. Label Encoding on Categorical features.
   ii. Normalizing data with min-max normalization
2. Stratified Sampling - Splitting Training and Testing data into 2/3rd and 1/3rd.
3. DecisionTreeClassifier model with criterion 'entropy' & max_depth = 4.

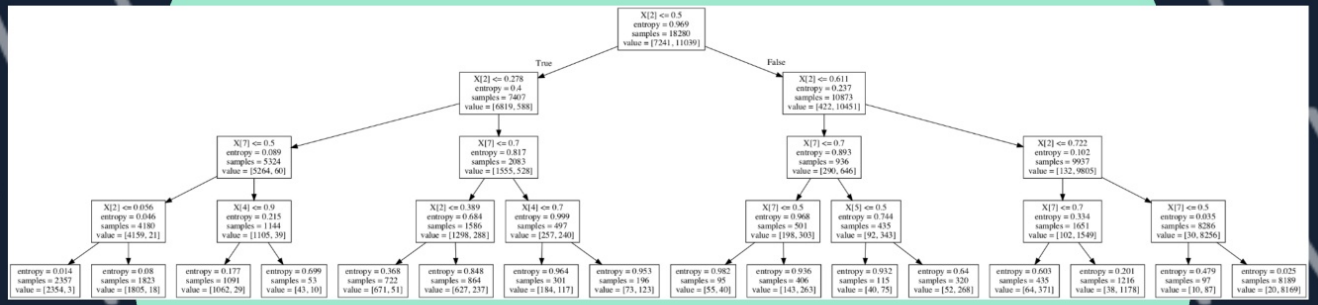
```
Shape after split:: (18280, 8) (9004, 8) (18280,) (9004,)
scoreTrain= 0.9483041575492341
scoreTest= 0.9461350510884051
```

Tree Result

Prediction Results

K-Fold Cross Validation

Results


X[2] <= 0.5
entropy = 0.969
samples = 18280
value = [7241, 11039]
True
False
X[2] <= 0.278
entropy = 0.4
samples = 7407
value = [6819, 588]
X[2] <= 0.611
entropy = 0.237
samples = 10873
value = [422, 10451]
X[7] <= 0.5
entropy = 0.089
samples = 5324
value = [5264, 60]
X[7] <= 0.7
entropy = 0.817
samples = 2083
value = [1555, 528]
X[7] <= 0.7
entropy = 0.893
samples = 936
value = [290, 646]
X[2] <= 0.722
entropy = 0.102
samples = 9957
value = [132, 9805]
X[2] <= 0.056
entropy = 0.046
samples = 4180
value = [4159, 21]
X[4] <= 0.9
entropy = 0.215
samples = 1144
value = [1105, 39]
X[2] <= 0.389
entropy = 0.684
samples = 1586
value = [1298, 288]
X[4] <= 0.7
entropy = 0.999
samples = 497
value = [257, 240]
X[7] <= 0.5
entropy = 0.968
samples = 501
value = [198, 303]
X[5] <= 0.5
entropy = 0.744
samples = 435
value = [92, 343]
X[7] <= 0.7
entropy = 0.334
samples = 1651
value = [102, 1549]
X[7] <= 0.5
entropy = 0.035
samples = 8286
value = [30, 8256]
entropy = 0.014
samples = 2357
value = [2354, 3]
entropy = 0.08
samples = 1823
value = [1805, 18]
entropy = 0.177
samples = 1091
value = [1062, 29]
entropy = 0.699
samples = 53
value = [43, 10]
entropy = 0.368
samples = 722
value = [671, 51]
entropy = 0.848
samples = 864
value = [627, 237]
entropy = 0.964
samples = 301
value = [184, 117]
entropy = 0.953
samples = 196
value = [73, 123]
entropy = 0.982
samples = 95
value = [55, 40]
entropy = 0.936
samples = 406
value = [143, 263]
entropy = 0.932
samples = 115
value = [40, 75]
entropy = 0.64
samples = 320
value = [52, 268]
entropy = 0.603
samples = 435
value = [64, 371]
entropy = 0.201
samples = 1216
value = [38, 1178]
entropy = 0.479
samples = 97
value = [10, 87]
entropy = 0.025
samples = 8189
value = [20, 8169]

Prediction Results

1. Decision Trees with 'Entropy' and 'Gini' criterion.
2. Tree depth levels ranging from 1 to 10.
3. Compare Training and Testing Accuracies for each depth.

K-Fold Cross Validation on Decision Tree Model

- Import cross_val_score, cross_val_predict from sklearn.model_selection.

- Perform 10-fold Cross Validation

```
1 from sklearn.model_selection import cross_val_score, cross_val_predict
2 from sklearn import metrics
3
4 # Perform 10-fold cross validation
5 scores = cross_val_score(dct, X, y, cv=10)
6 print("Cross-validated scores:", scores)
```

```
Cross-validated scores: [0.89446684 0.93001099 0.92085013 0.92927812 0.93843899 0.92671308
 0.92192082 0.88489736 0.92519252 0.85588559]
```

Naive Bayes

It is a classification technique based on bayes theorem.

It assumes that features are independent.

Posterior= ((likelihood)*(prior probability of proposition))/(prior probability of evidence)

$P(A|B) = [P(B|A) * P(A)]/P(B)$

Prior= P(A), P(B) it is marginal probability distribution representing knowledge or uncertainty of data object prior or before observing it.

Posterior=P(A| B) it is conditional probability distribution representing what parameters are likely after observing the data object.

Likelihood=p(B| A) it is the conditional probability of falling under specific category or class.

Title

Title

Title

Title

Title

Title

Steps:

Convert data into frequency tables.

Finding likelihood.

Posterior probability is calculated for each class.

scikit-learn library

Gaussian

Multinomial

Bernoulli

Code

```
from sklearn.naive_bayes import MultinomialNB

Nb_model = MultinomialNB()

Nb_model.fit(X_train,y_train)

MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

Alpha :smoothing parameter

Class_prior:If specified the priors are not adjusted according to the data.

Fit_prior:Whether to learn class prior probabilities or not. If false, a uniform prior will be used.

Accuracy

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,pred)
```

82.35%

Confusion Matrix

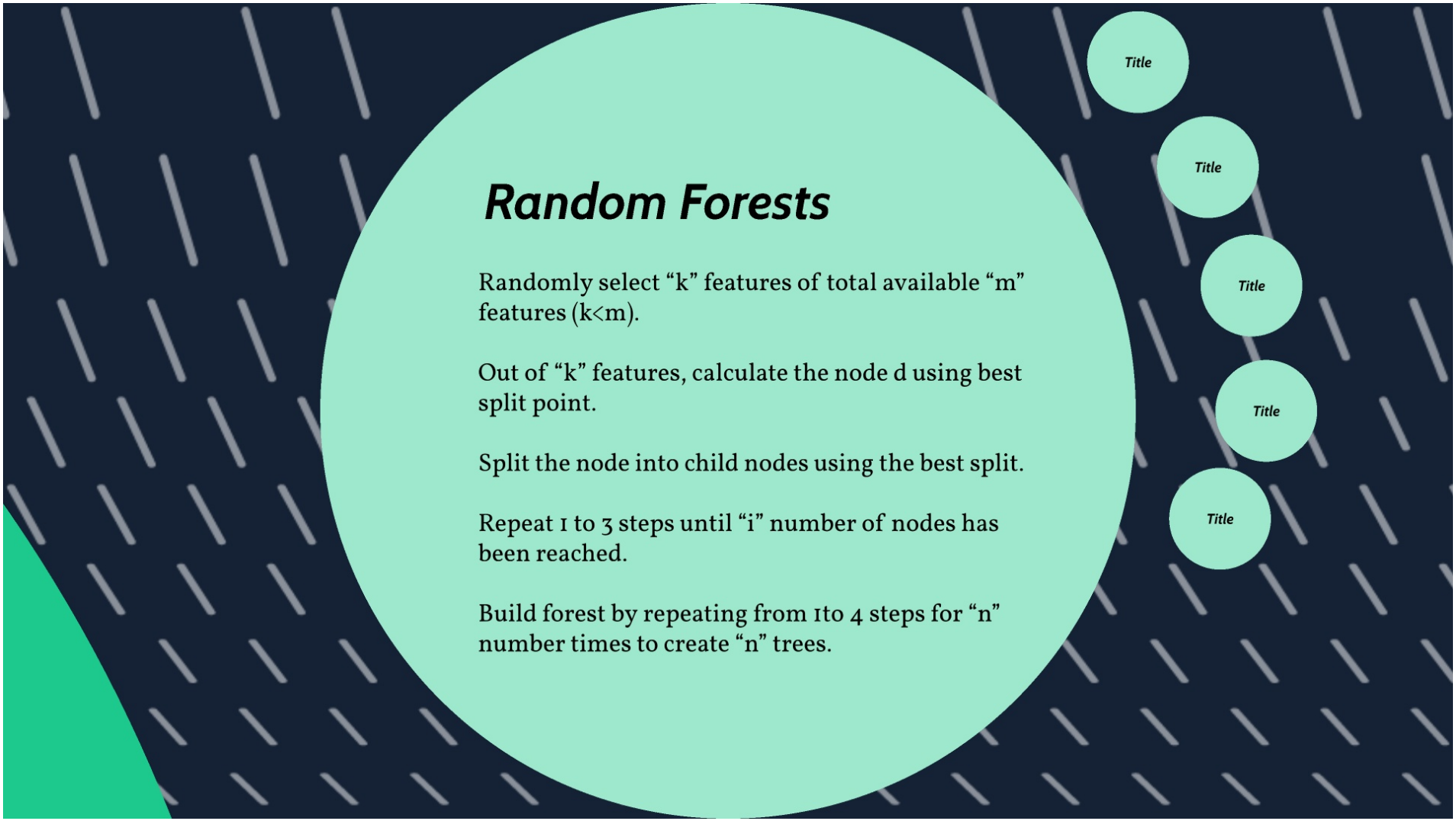
```
array([[2040, 1243],
       [ 201, 4701]])
```

Cross validation

```
from sklearn.model_selection import cross_val_score

score=cross_val_score(Nb_model, X, y, cv=10,
scoring='accuracy')
score.mean()

0.8221979169532062
```


Title
Title
Title
Title
Title
Random Forests
Randomly select “k” features of total available “m” features (k<m).
Out of “k” features, calculate the node d using best split point.
Split the node into child nodes using the best split.
Repeat 1 to 3 steps until “i” number of nodes has been reached.
Build forest by repeating from 1to 4 steps for “n” number times to create “n” trees.

Prediction

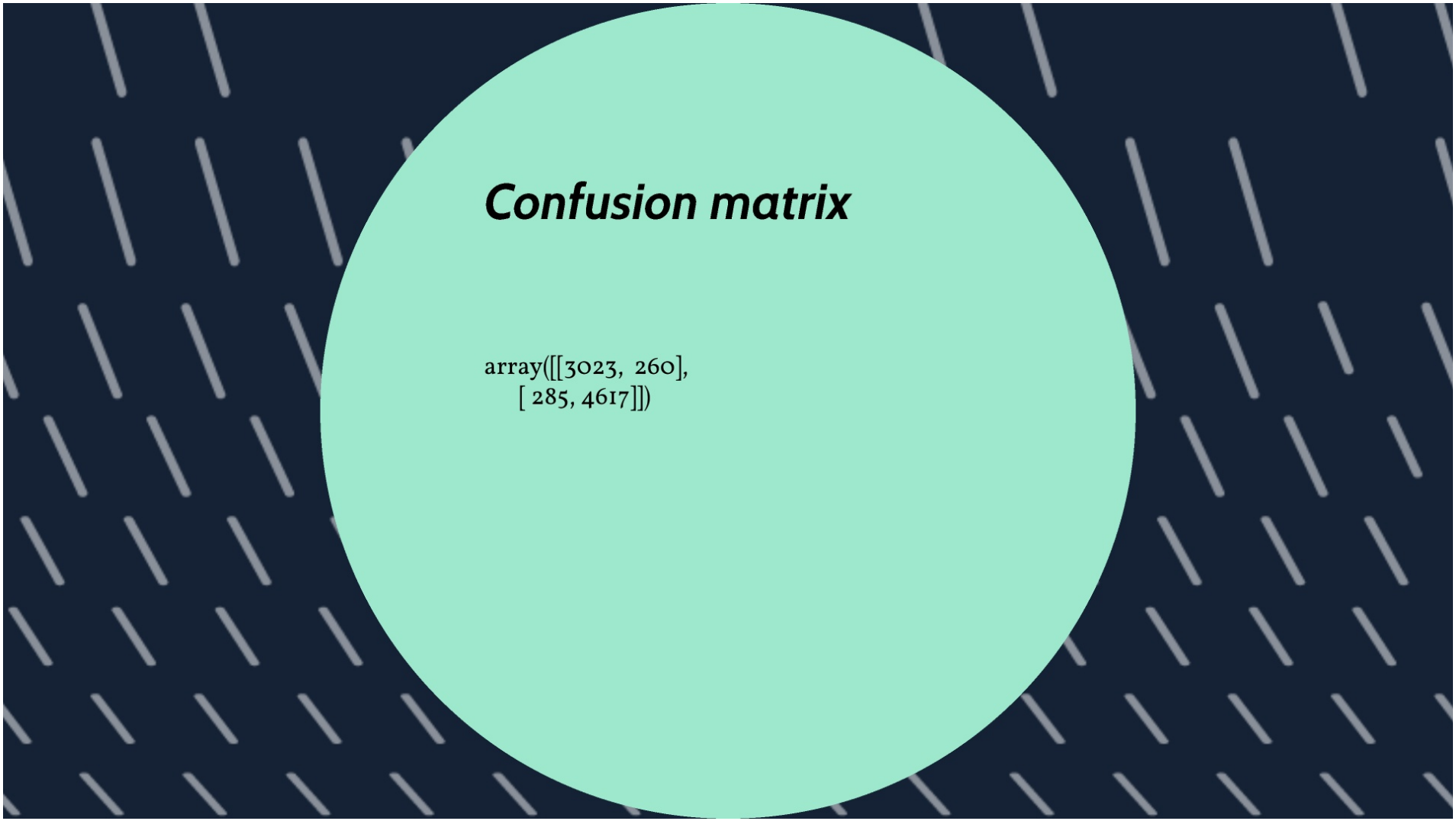
Take the test features and predict the outcome using rules of each randomly created tree and save the predicted target.

Calculate the votes for each predicted target. Consider the high voted predicted target as the final prediction from the random forest (majority voting).

Code

```
from sklearn.ensemble import RandomForestClassifier
RF_model =
RandomForestClassifier(n_estimators=10,random_state=42,bootstrap
=True)
RF_model.fit(X_train, y_train)
RF_model.score(X_test, y_test)
```

0.93341478313989


Confusion matrix
array([[3023, 260],
[285, 4617]])

Cross validation

```
from sklearn.model_selection import cross_val_score
score=cross_val_score(RF_model, X, y, cv=10,
scoring='accuracy')
score.mean()
```

```
0.9364805869951329
```

Advantages

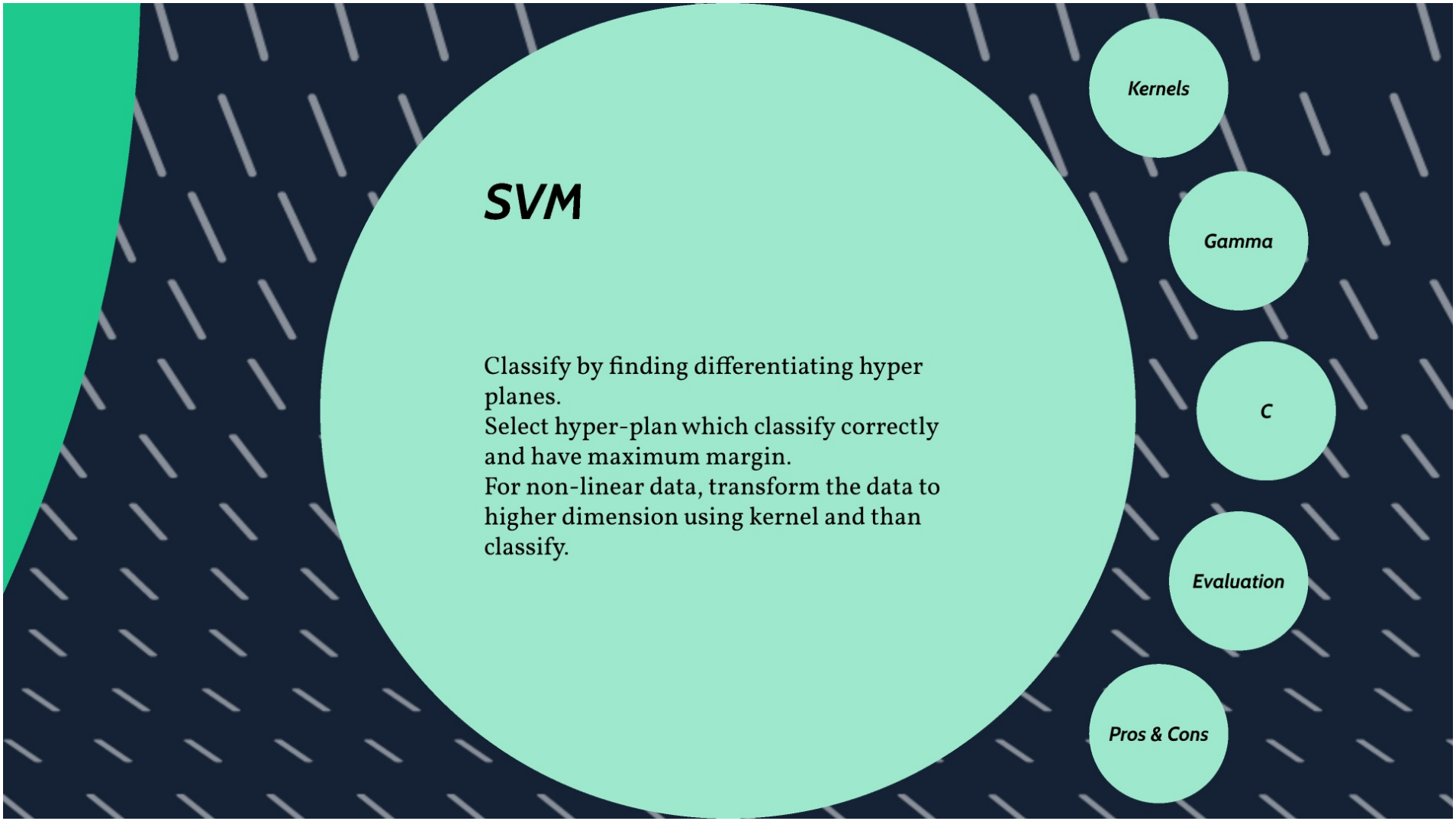
It can be used for feature engineering i.e. identifying the most important features out of available features from training data set.

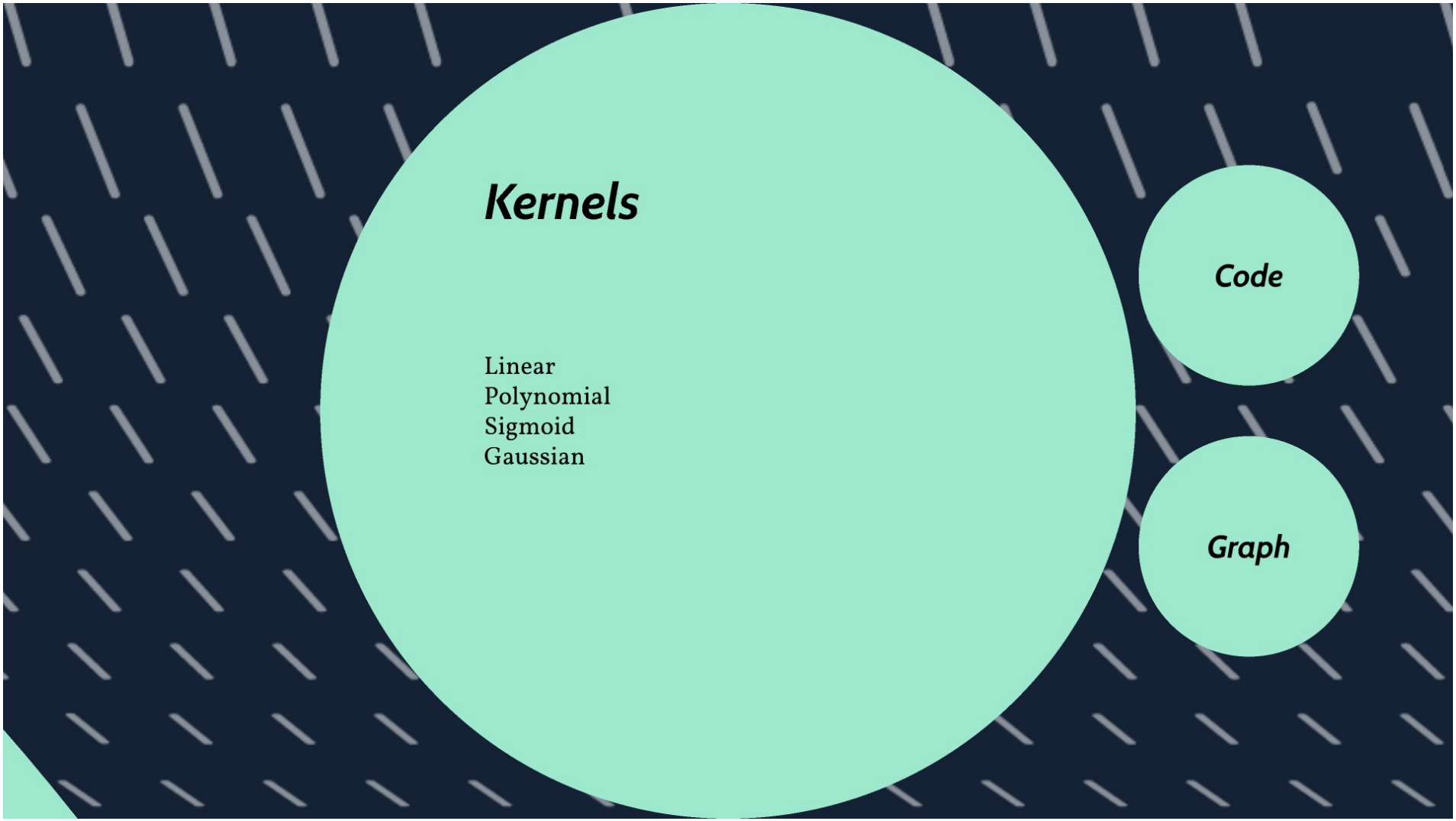
Overfitting never occurs when we use random forest in classification problem.

It has an effective method to find missing data and maintains accuracy when large proportion of the data are missing.

No need to scale data. (normalization, standardization)
Robust to outliers.

Very stable even new point is introduced in the dataset as it might affect one tree and doesn't impact all the trees.


SVM
Classify by finding differentiating hyper planes.
Select hyper-plan which classify correctly and have maximum margin.
For non-linear data, transform the data to higher dimension using kernel and than classify.
Kernels
Gamma
C
Evaluation
Pros & Cons

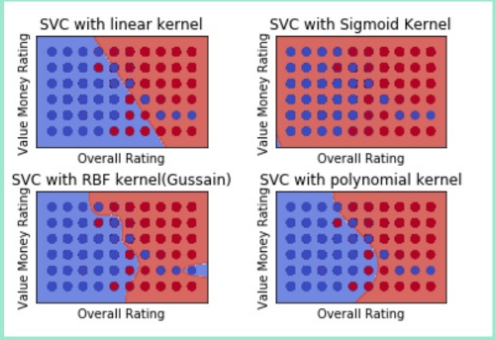

Kernels
Linear
Polynomial
Sigmoid
Gaussian
Code
Graph

Example

```
from sklearn.svm import SVC
sig_svclassifier = SVC(kernel='sigmoid')
sig_svclassifier.fit(X_train, y_train)
sig_y_pred = sig_svclassifier.predict(X_test)

from sklearn.svm import SVC
g_svclassifier = SVC(kernel='rbf')
g_svclassifier.fit(X_train, y_train)
g_y_pred = g_svclassifier.predict(X_test)
```

Kernel Graphs



Gamma

Kernel coefficient for ‘rbf’, ‘poly’ and ‘sigmoid’.

Higher the value of gamma, will try to exact fit as per training data set i.e. generalization error and cause over-fitting problem.

small gamma will give you low bias and high variance and vice versa.

Code

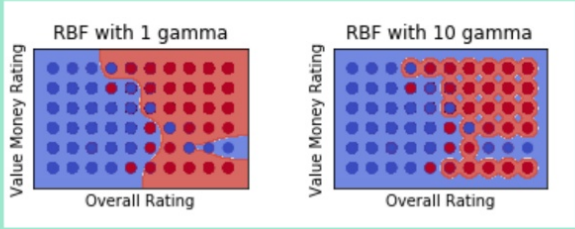
Graph

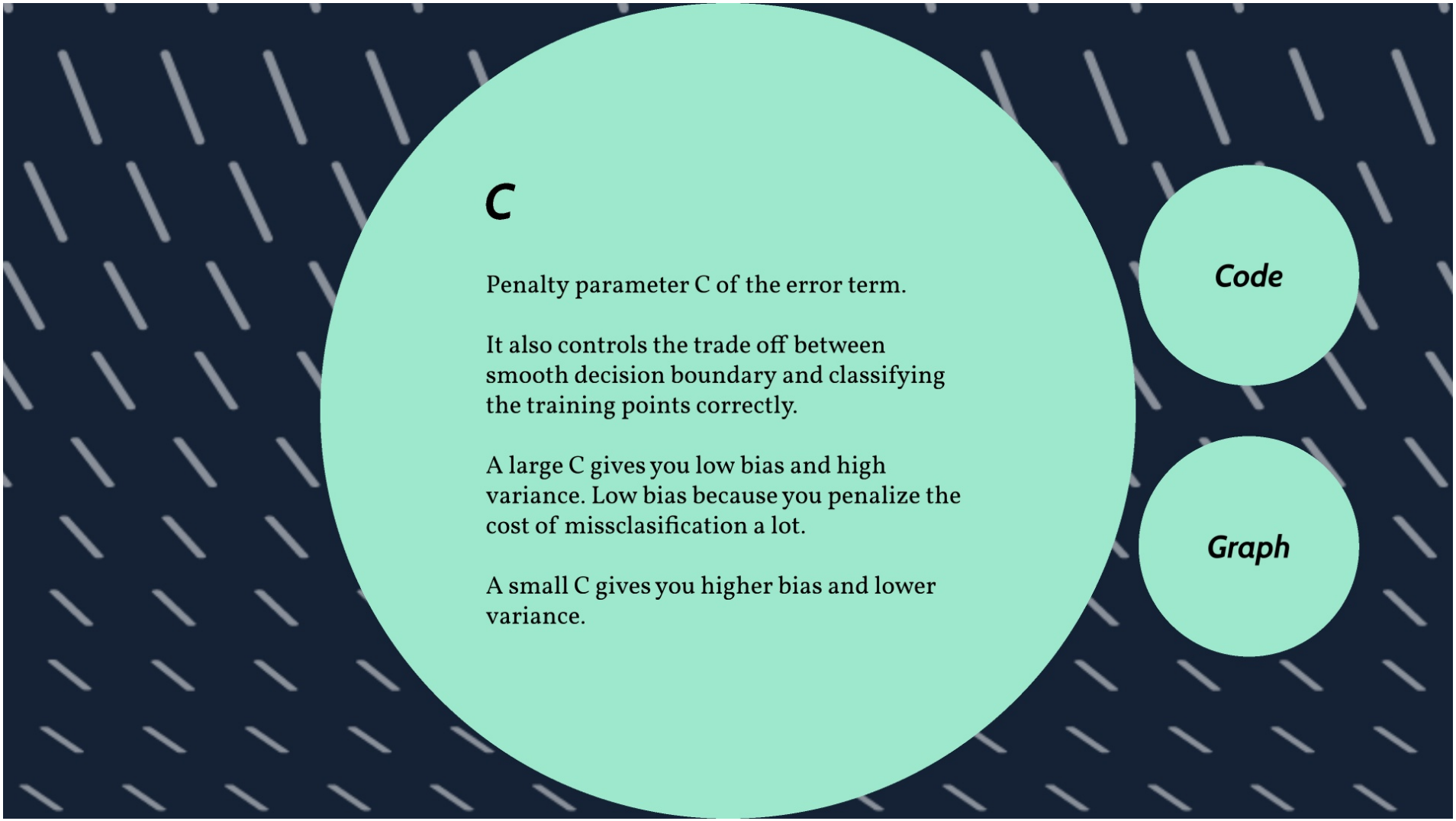
Example

```
rbf_svc = svm.SVC(kernel='rbf', gamma=1,
C=C).fit(X, y)

rbf_svc = svm.SVC(kernel='rbf', gamma=10,
C=C).fit(X, y)
```

Graph


RBF with 1 gamma
Value Money Rating
Overall Rating
RBF with 10 gamma
Value Money Rating
Overall Rating


C
Penalty parameter C of the error term.
It also controls the trade off between smooth decision boundary and classifying the training points correctly.
A large C gives you low bias and high variance. Low bias because you penalize the cost of missclasification a lot.
A small C gives you higher bias and lower variance.
Code
Graph

Example

```
svc = svm.SVC(kernel='rbf', gamma=0.7,
C=1).fit(X, y)

rbf_svc = svm.SVC(kernel='rbf', gamma=0.7,
C=10).fit(X, y)

poly_svc = svm.SVC(kernel='rbf',
gamma=0.7, C=100).fit(X, y)
```

Graph


RBF with 1 C
Value Money Rating
Overall Rating
RBF with 10 C
Value Money Rating
Overall Rating
RBF with 100 C
Value Money Rating
Overall Rating

Evaluation

Linear Kernel Performance

	precision	recall	f1-score	support
0	0.92	0.93	0.92	3253
1	0.95	0.95	0.95	4932
accuracy			0.94	8185

Polynomial Kernel Performance

	precision	recall	f1-score	support
0	0.92	0.93	0.92	3253
1	0.96	0.94	0.95	4932
accuracy			0.94	8185

Sigmoid Kernel Performance

	precision	recall	f1-score	support
0	0.00	0.00	0.00	3253
1	0.43	0.51	0.47	4932
accuracy			0.31	8185

Pros and Cons

Pros:

It works really well with clear margin of separation.

It is effective in high dimensional spaces.

Cons:

It doesn't perform well, when we have large data set because the required training time is higher.

It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping.


Introduction
Classification
Clustering
Thank You!
Data Mining on Airline Reviews Dataset