Data Preprocessing

first i set option to display all columns
for less memory usage and faster training i converted dtype from float64 to float32 and int64 to int8
checking for any missing value
checking for any duplicated row
deleting single-valued column bcz it will not contribute anything to model prediction
checking min and max value of each column
separating output labels from features

Identifying and Handling Outliers

```
Outliers are very big problem in this dataset
Some of columns contain values in the order from e+0 to e+9 which will definitely
affect model performance
```

first i tried IQR method but it identifies most of rows as outlier so this is not good method here then

```
i took log of columns which contain very extreme values
taking log(1+x) so that very small values does not get converted to -infinity
```

Robust Scaling

```
it handles outliers by scaling based on median and IQR
```

Visualizations

Class distributions

Now biggest problem is that data is highly imbalanced, about only 3% cases are bankrupt so we will not prefer accuracy much here
![Class Distributions Plot][images/class_distributions.png]

Correlation Heatmap

hello this is correlation heatmap

```
splitting dataset into training and testing set
```

SMOTE (Synthetic minority oversampling technique)

```
In this imbalanced dataset we have two options either we can use class-
weight="balanced" which works in most of the models or we can balance dataset by
oversampling minority class
```

Now we train various models on this preprocessed and balanced dataset

Logistic Regression

trying different solvers with l2 and l1

```
liblinear- for smaller datasets, Good for binary classification and supports L1
and L2 regularization

saga- larger datasets or sparse data, Works with L1, L2, and elasticnet, supports
multiclass

lbfgs- good default, optimized for l2, better for larger data than liblinear
```

(default) solver=lbfgs, penalty=l2

```
              precision    recall  f1-score   support
           0       0.99      0.88      0.94      1324
           1       0.18      0.82      0.29        40

Confusion Matrix:[[1171  153]
                 [   7   33]]
ROC AUC Score: 0.93
PR AUC score: 0.30
Accuracy_score: 0.88
best_f1: 0.42
best_thresh: 0.81
```

solver="liblinear",penalty="l2"

```
              precision    recall  f1-score   support
           0       0.99      0.88      0.94      1324
           1       0.17      0.80      0.28        40

Confusion Matrix:  [[1170  154]
                   [   8   32]]
ROC AUC Score: 0.93
PR AUC score: 0.30
```

```
Accuracy_score: 0.88
best_f1: 0.41
best_thresh: 0.80
```

solver="saga",penalty="l1"

```
              precision    recall  f1-score   support
           0       0.99      0.88      0.94      1324
           1       0.17      0.80      0.28        40

Confusion Matrix:  [[1171  153]
                    [   8   32]]
ROC AUC Score: 0.93
PR AUC score: 0.30
Accuracy_score: 0.88
best_f1: 0.42
best_thresh: 0.81
```

solver="liblinear",penalty="l1"

```
              precision    recall  f1-score   support
           0       0.99      0.89      0.94      1324
           1       0.18      0.78      0.29        40

Confusion Matrix:  [[1179  145]
                    [   9   31]]
ROC AUC Score: 0.93
PR AUC score: 0.31
Accuracy_score: 0.89
best_f1: 0.38
best_thresh: 0.86
```

best_f1 for logistic regression is 0.42
best model= (default) solver=lbfgs, penalty=l2

Regularization

```
C:[0.01,0.1,1,5,10]

best_f1: 0.30, best_thresh: 0.64
best_f1: 0.37, best_thresh: 0.72
best_f1: 0.41, best_thresh: 0.80
best_f1: 0.39, best_thresh: 0.78
best_f1: 0.40, best_thresh: 0.77
```

```
C: [0.5,1,1.5,2,2.5,3]

best_f1: 0.41, best_thresh: 0.77
best_f1: 0.41, best_thresh: 0.80
best_f1: 0.42, best_thresh: 0.81
best_f1: 0.41, best_thresh: 0.81
best_f1: 0.41, best_thresh: 0.82
best_f1: 0.40, best_thresh: 0.82
```

best regularization strength= 1.5

```
              precision    recall  f1-score   support
           0       0.99      0.89      0.94      1324
           1       0.17      0.78      0.28        40

Confusion Matrix:  [[1176  148]
                    [   9   31]]
ROC AUC Score: 0.93
PR AUC score: 0.30
Accuracy_score: 0.88
best_f1: 0.42
best_thresh: 0.81
```

but our precision and recall are lower than default one, thus we will stick with default solver=lbfgs, penalty=l2, C=1

Support Vector Machines

Training on imbalanced dataset

```
svm_clf=SVC(
    kernel="rbf",
    probability=True,
    class_weight="balanced",
    C=1.0,
    gamma="scale"
)

               precision    recall  f1-score   support
           0       0.98      0.66      0.79      1324
           1       0.05      0.57      0.09        40

Confusion Matrix:  [[872 452]
                    [ 17  23]]
ROC AUC Score: 0.73
PR AUC score: 0.10
```

```
Accuracy_score: 0.66
best_f1: 0.17
best_thresh: 0.06
```

Here SVM tries to draw a boundary that maximizes margin but with very few minority class samples it heavily favours majority class

On balanced dataset

```
              precision    recall  f1-score   support
           0       0.98      0.76      0.86      1324
           1       0.06      0.50      0.11        40

Confusion Matrix:  [[1006  318]
                   [  20   20]]
ROC AUC Score: 0.71
PR AUC score: 0.06
Accuracy_score: 0.75
best_f1: 0.13
best_thresh: 0.83
```

Here balanced dataset performed better but after threshold optimizing imbalanced dataset has better score, but we would focus on balanced dataset bcz it gives a more balanced decision boundary and also helpful for recall

GridSearchCV

```
params={
    "C":[0.1,0.5,1,2],
    "kernel":["rbf"],
    "gamma":["scale"]
}
SVC(C=1, class_weight='balanced', probability=True, random_state=42)

{'C': 1, 'gamma': 'scale', 'kernel': 'rbf'}
grid.best_score_= 0.096
              precision    recall  f1-score   support
           0       0.98      0.66      0.79      1324
           1       0.05      0.57      0.09        40

Confusion Matrix:  [[872 452]
                   [ 17  23]]
ROC AUC Score: 0.73
PR AUC score: 0.10
Accuracy_score: 0.66
best_f1: 0.16
best_thresh: 0.07
```

In this case, GridSearchCV has chosen model with C=1, means no regularization

Decision Tree

(default) max_depth=None

```
              precision    recall  f1-score   support
           0       0.98      0.95      0.97      1324
           1       0.23      0.45      0.31        40

Confusion Matrix:  [[1264   60]
                    [  22   18]]
ROC AUC Score: 0.70
PR AUC score: 0.12
Accuracy_score: 0.94
best_f1: 0.31
best_thresh: 0.01
```

```
max_depth=3, best_f1: 0.35
             best_thresh: 0.82
max_depth=5, best_f1: 0.36
             best_thresh: 0.94
max_depth=6, best_f1: 0.38
             best_thresh: 0.94
max_depth=7, best_f1: 0.36
             best_thresh: 0.96
max_depth=10,best_f1: 0.33
             best_thresh: 0.51
```

best_f1 for decision tree is 0.38
best model= (max_depth=6)

RandomForestClassifier

(max_depth=6)

```
n_estimators=100, best_f1: 0.48
                  best_thresh: 0.84
```

```
n_estimators=150, best_f1: 0.49
                  best_thresh: 0.84

              precision    recall  f1-score   support
```

```
           0       0.99      0.92      0.95      1324
           1       0.22      0.78      0.34        40

Confusion Matrix: [[1213  111]
                   [   9   31]]
ROC AUC Score: 0.94
PR AUC score: 0.42
Accuracy_score: 0.91
best_f1: 0.49
best_thresh: 0.85
```

```
n_estimators=200, best_f1: 0.48
                  best_thresh: 0.86
```

(max_depth=8, n_estimators=150)

```
              precision    recall  f1-score   support
           0       0.99      0.94      0.96      1324
           1       0.26      0.75      0.38        40

Confusion Matrix:  [[1238   86]
                    [  10   30]]
ROC AUC Score: 0.94
PR AUC score: 0.42
Accuracy_score: 0.93
best_f1: 0.48
best_thresh: 0.81
```

```
best_f1 for RandomForestClassifier is 0.48
best model= (max_depth=6, n_estimators=150)
```

GridSearchCV

MY MISTAKE HERE- Since SMOTE is applied before and cross-validation is used in GridSearch, we would train on synthetic data which is easier and does not represent real world performance thus

We perform GridSearchCV on dataset with (No SMOTE)

```
params={
    "max_depth":[6,8,10],
    "max_features":["sqrt","log2"],
    "n_estimators":[100,150,200],
    "bootstrap":[True],
}

{'bootstrap': True, 'max_depth': 8, 'max_features': 'log2', 'n_estimators': 150}
```

```
grid.best_score_= 0.45
              precision    recall  f1-score   support
           0       0.99      0.94      0.97      1324
           1       0.28      0.72      0.40        40

Confusion Matrix: [[1249   75]
                   [  11   29]]
ROC AUC Score: 0.94
PR AUC score: 0.45
Accuracy_score: 0.94
best_f1: 0.56
best_thresh: 0.75
```

Again with wider parameter space

```
params={
    "max_depth":[6,8,10],
    "max_features":["sqrt","log2"],
    "n_estimators":[150,200],
    "bootstrap":[True],
    "min_samples_split":[10,20,40],
    "min_samples_leaf":[5,10,15]
}

{'bootstrap': True, 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf':
10, 'min_samples_split': 20, 'n_estimators': 200}
grid.best_score_:  0.479
              precision    recall  f1-score   support
           0       0.99      0.95      0.97      1324
           1       0.32      0.68      0.44        40

Confusion Matrix:  [[1267   57]
                    [  13   27]]
ROC AUC Score: 0.94
PR AUC score: 0.46
Accuracy_score: 0.95
best_f1: 0.54
best_thresh: 0.66
```

```
this time results were lower but we have used min_samples_split and
min_samples_leaf
```

Again with slightly different parameter space

```
params={
    "max_depth":[8],
```

```
    "max_features":["sqrt","log2"],
    "n_estimators":[150,200],
    "bootstrap":[True],
    "min_samples_split":[15,20,25,30],
    "min_samples_leaf":[7,10,13]
}

{'bootstrap': True, 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 7,
'min_samples_split': 15, 'n_estimators': 150}
grid.best_score_= 0.47
              precision    recall  f1-score   support
           0       0.99      0.96      0.97      1324
           1       0.32      0.68      0.44        40
Confusion Matrix:  [[1267   57]
                    [  13   27]]
ROC AUC Score: 0.94
PR AUC score: 0.49
Accuracy_score: 0.95
best_f1: 0.52
best_thresh: 0.66
```

Extracting top features from our best RandomForest model based on
1) top 30 and 2) features which contribute 95% of whole

```
top features (95%) count = 20
Creating new training and testing set based on top 30 and 20 features
```

XGBoostClassifier

```
n_estimators=500,
learning_rate=0.05,
max_depth=5,
scale_pos_weight=scale_pos_weight,
eval_metric="aucpr",
early_stopping_rounds=30

             precision    recall  f1-score   support
           0       0.99      0.94      0.97      1324
           1       0.27      0.70      0.39        40

Confusion Matrix: [[1249   75]
                   [  12   28]]
ROC AUC Score: 0.91
PR AUC score: 0.39
Accuracy_score: 0.94
best_f1: 0.45
best_thresh: 0.81
```

GridSearchCV

```
params={
    'n_estimators':[300],
    'max_depth':[3,5,7],
    'gamma':[0,0.1,0.3],
    'learning_rate':[0.01,0.05,0.1],
    'min_child_weight':[1,5,10]
}

{'gamma': 0, 'learning_rate': 0.1, 'max_depth': 5, 'min_child_weight': 5,
'n_estimators': 300}
grid.best_score_= 0.51
              precision    recall  f1-score   support
           0       0.99      0.98      0.98      1324
           1       0.45      0.53      0.48        40

Confusion Matrix:  [[1298   26]
                   [  19   21]]
ROC AUC Score: 0.93
PR AUC score: 0.44
Accuracy_score: 0.97
best_f1: 0.54
best_thresh: 0.40
```

Again GridSearchCV

```
params={
    'n_estimators':[300],
    'max_depth':[4,5,6],
    'gamma':[0,0.2],
    'learning_rate':[0.075,0.1,0.125],
    'subsample':[0.8,1],
    'colsample_bytree':[0.8,1],
    'min_child_weight':[5,7]
}

{'colsample_bytree': 0.8, 'learning_rate': 0.125, 'max_depth': 6,
'min_child_weight': 7, 'n_estimators': 300, 'subsample': 1}
grid.best_score_= 0.525
              precision    recall  f1-score   support
           0       0.99      0.98      0.98      1324
           1       0.46      0.55      0.50        40

Confusion Matrix:  [[1298   26]
                   [  18   22]]
ROC AUC Score: 0.93
PR AUC score: 0.41
Accuracy_score: 0.97
```

```
best_f1: 0.53
best_thresh: 0.40
```

this time best_f1 after threshold optimization and PR AUC score are lower than previous one 0.54 so We will choose best model from previous grid search

With top 20 features performance of best XGBoost model go down

With top 30 features performance of best XGBoost model also lower

Ensemble Methods

Voting Classifier

i have tried first hard voting which is very very bad so we will only use voting=soft for prediction based on probabilities

i included first best decision tree along with logistic,randomforest,xgboost but it worsens result

```
Logistic(max_iter=6000,solver="lbfgs",penalty="l2",C=1)
RandomForestClassifier(n_estimators=150,max_features="log2",max_depth=8,bootstrap=
True)
XGBClassifier(n_estimators= 300,gamma= 0, learning_rate= 0.1,max_depth=
5,min_child_weight= 5,scale_pos_weight=scale_pos_weight)

                precision    recall  f1-score   support
           0       0.97      1.00      0.99      1324
           1       0.50      0.15      0.23        40
Confusion Matrix:  [[1318    6]
                    [  34    6]]
ROC AUC Score: 0.93
PR AUC score: 0.40
Accuracy_score: 0.97
best_f1: 0.52
best_thresh: 0.26
```

Stacking Classifier