universidade de aveiro

deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

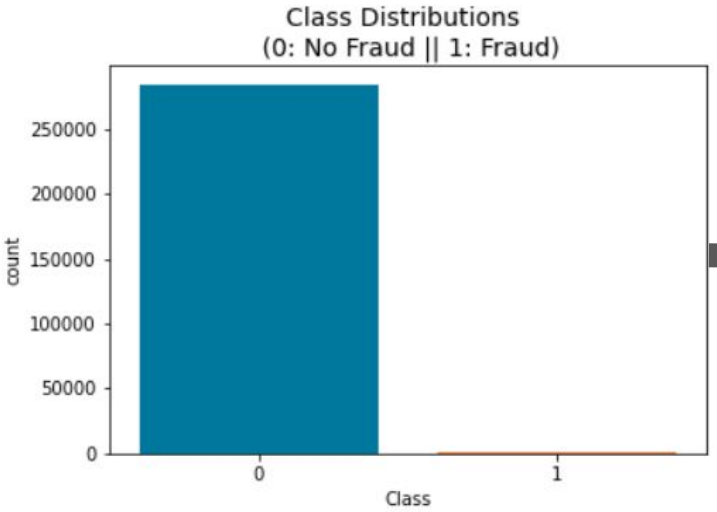
Credit Card Fraud Detection

Henrique Sousa, 98324
Vasco Regal, 96736

Context

- It’s important to make sure people don’t get charged for items they did not purchase
- We will use an unbalanced dataset available on Kaggle
- We want to implement and compare different machine learning models:
  - Logistic Regression
  - Support Vector Machines
  - Naive Bayes
  - Neural Networks

Dataset Preparation

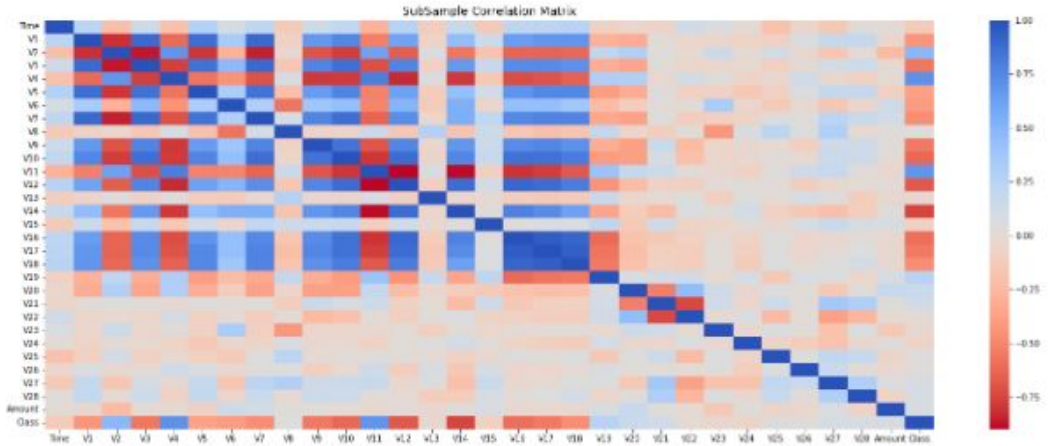


284,315 non-fraud & 492 Fraud

492 non-fraud & 492 Fraud

Dataset Preparation

- **Random Under-Sampling**
  - Randomly choose 492 non-fraud cases and use the 492 fraud ones.
- **Correlation Matrix**
  -



Dataset Preparation

- **Anomaly Detection**
  - Interquartile Range Method = Q3 - Q1
  - lower limit = Q1 - 1.5*IQR
  - upper limit = Q3 + 1.5*IQR
  - Everything above the upper limit or below the lower limit will be considered an outlier and thus will be removed

Models

- We tested every model with:
  - Base model
  - Hyperparameter tuning
  - K-Fold Cross-Validation on the hypertuned model.

- The results vary for each different model used.

Model Testing

```python
def train_and_analyze(model, X_train, y_train, X_test, y_test):
    model.fit(X_train, y_train)

    print(f"> Model: {model.__class__}\n")

    X_test_prediction = model.predict(X_test)
    test_data_accuracy = accuracy_score(X_test_prediction, y_test)
    f1 = f1_score(y_test, X_test_prediction)

    print()
    print("---- Model Analysis ----")
    print('Accuracy: ', test_data_accuracy)
    print('F1 Score: ', f1)

    print()

    print("Confusion Matrix: ")
    cm = confusion_matrix(y_test, model.predict(X_test))

    fig, ax = plt.subplots(figsize=(8, 8))
    ax.imshow(cm)
    ax.grid(False)
    ax.xaxis.set(ticks=(0, 1), ticklabels=('Predicted 0s', 'Predicted 1s'))
    ax.yaxis.set(ticks=(0, 1), ticklabels=('Actual 0s', 'Actual 1s'))
    ax.set_ylim(1.5, -0.5)
    for i in range(2):
        for j in range(2):
            ax.text(j, i, cm[i, j], ha='center', va='center', color='red')
    plt.show()

    print()
    print("---- Classification Report ----")
    print(classification_report(y_test, model.predict(X_test)))
```

Hyper-Parameter Finding

```
def hyperparameters(model, params, X, y):
    """
    Find hyperparameters for a model
    """
    print("> Hyper Parameter Tuning")
    print("Finding Best Params for Model ", model.__class__)
    model = GridSearchCV(model, params, scoring="accuracy")
    model.fit(X, y)
    print(" Best Params: ")
    print(model.best_params_)

    return model.best_params_
```

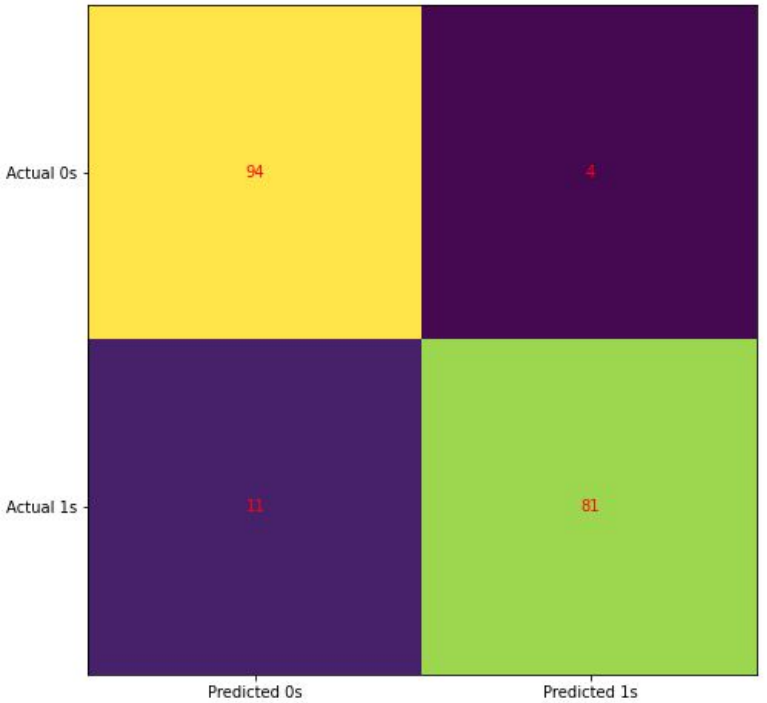
Cross Validation Score And K-Fold Best Estimator

- With K=10

```
def kfold(model, k, X_train, y_train, X_test, y_test):
    kf = KFold(n_splits=k, shuffle=True)
    scores = cross_validate(model, X_train, y_train, scoring="accuracy", cv=k, return_estimator=True)
    return scores["estimator"][np.argmax(scores["test_score"])]
```

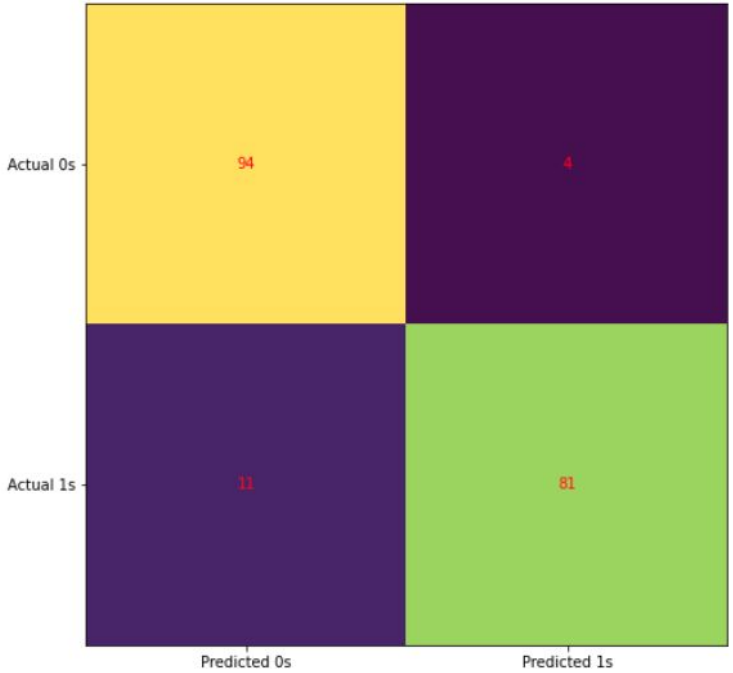
Logistic Regression

	Base	HyperTuned	K-fold CV
Accuracy	0.921	0.921	0.921
F1 Score	0.915	0.915	0.915



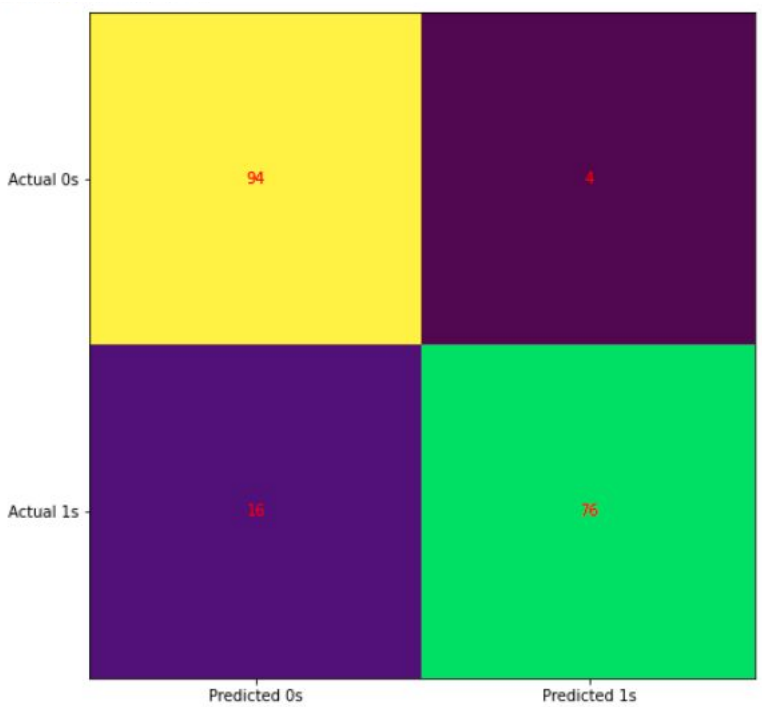
Support Vector Machines

	Base	HyperTuned	K-fold CV
Accuracy	0.911	0.915	0.915
F1 Score	0.907	0.912	0.912



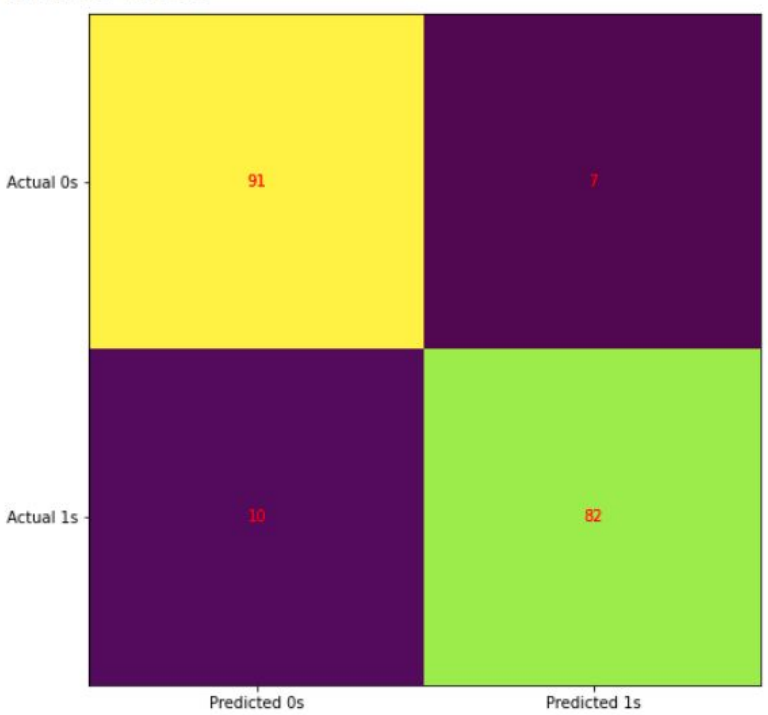
Naive Bayes

	Base	HyperTuned	K-fold CV
Accuracy	0.895	0.895	0.895
F1 Score	0.884	0.884	0.884

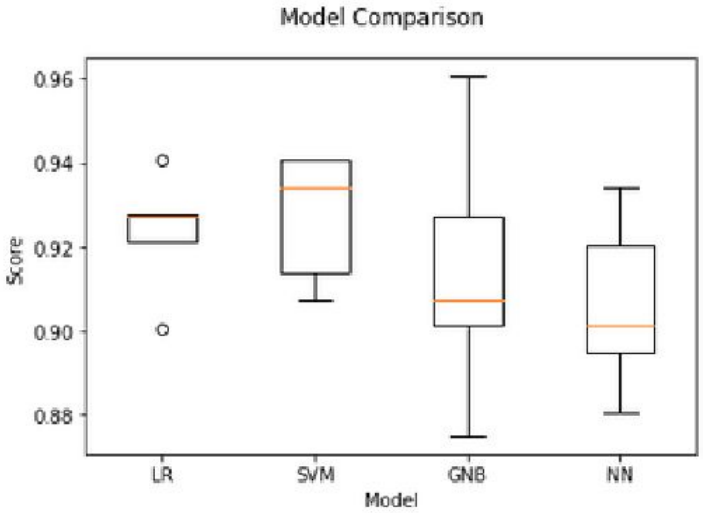


Neural Networks

	Base	HyperTuned	K-fold CV
Accuracy	0.900	0.910	0.911
F1 Score	0.895	0.901	0.910



Model Comparison



Comparison With Other Works

- Credit Fraud - Dealing with Imbalanced Datasets, *Janio Martinez Bachmann*
- Credit card dataset: SVM Classification, *Pierre-Alexis LE BORGNE*
- Automated Hyperparameter Tuning, *Pavan Sanagapati*